UNIVERSITÉ DE MONTRÉAL

ÉVALUATION DE LA TECHNIQUE D'ANALYSE SÉMANTIQUE LATENTE POUR LA CORRECTION D'ANALYSE DE CAS

ABOUELFOUTOUH ABDELILAH
GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
JUILLET 2006

Your file Votre référence
ISBN: 978-0-494-19275-7
Our file Notre référence
ISBN: 978-0-494-19275-7

UNIVERSITÉ DE MONTRÉAL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

ÉVALUATION DE LA TECHNIQUE D'ANALYSE SÉMANTIQUE LATENTE POUR LA CORRECTION D'ANALYSE DE CAS

présenté par : ABOUELFOUTOUH Abdelilah
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen constitué de :

M. GAGNON Michel, Ph.D., président
M. DESMARAIS Michel, Ph.D, membre et directeur de recherche
M. PESANT Gilles, Ph.D, membre

À la mémoire de mon père
Que Dieu l'entoure de sa miséricorde
À ma famille

Remerciements

Mes sincères remerciements vont à mon directeur de recherche, Monsieur Michel Desmarais pour avoir encadré ce mémoire avec beaucoup de compétence, pour sa disponibilité et pour avoir éclairé mon chemin tout au long de cette recherche.

Je remercie Messieurs Michel Gagnon et Gilles Pesant d'avoir accepté de juger ce travail.

Je tiens à remercier vivement mes amis et collègues de recherche à l'École polytechnique surtout Peyman et Alejandro, merci pour vos encouragements, conseils et longues discussions.

Enfin, pour leur soutien sans faille et leur présence, j'exprime ma profonde reconnaissance à toute ma famille.

Résumé

Si l'objectif de corriger automatiquement les travaux des étudiants a toujours été enchanteur pour les professeurs, la pratique de plus en plus courante chez les étudiants de déposer des copies électroniques de leurs travaux rend cet objectif encore plus attrayant. Plusieurs tentatives ont été faites pour prendre en charge cette routine ou, tout au moins, assister à son accomplissement. Les premières expériences datent des années soixante et, quarante ans après, cet objectif ambitieux est loin d'être complètement atteint. La correction de texte libre et les dissertations sont des défis de taille à relever.

La plupart des approches adoptées se basent surtout sur l'interprétation des traits superficiels tels que le nombre de mots, la ponctuation du texte ou encore la co-occurrence des termes. Ces approches ne permettent pas de comprendre la sémantique du texte. Plusieurs problèmes se posent. Premièrement la synonymie : plusieurs mots syntaxiquement différents sont sémantiquement équivalents pour exprimer le même concept, par exemple « voiture » et « automobile ». Une deuxième raison est la polysémie : un mot prend le sens de son contexte. Par exemple le mot « avocat » peut signifier le fruit de l'avocatier aussi bien que la profession d'avocat. Une troisième raison est que ces approches ne peuvent pas détecter au-delà des associations directes entre les mots, c'est-à-dire que les mots doivent apparaître ensemble (association directe) pour être considérés sémantiquement ou contextuellement liés. Or, se limiter à ce niveau d'association ne reflète pas la réalité car les mots similaires ou opposés apparaissent rarement dans le même contexte (répéter des synonymes causera une redondance et répéter des antonymes produira une incohérence).

D'autre part, la syntaxe et les associations directes ne sont pas suffisantes pour refléter la sémantique. Tant que ces approches n'arrivont pas à apprécier la dimension sémantique du texte, elles ne pourront pas en extraire les éléments d'une bonne ou une mauvaise réflexion, ainsi leur application dans la correction automatique ne serait pas d'une grande efficacité.

Dans ce mémoire, on utilise une technique orientée vers la sémantique des mots, appelée l'analyse sémantique latente (Latente Semantic Analysis, LSA), pour effectuer une correction automatique des textes. Cette technique est issue du domaine de la recherche d'information et apporte des réponses aux problèmes sémantiques jusque là irrésolus grâce à sa capacité d'extraire le sens des mots par leurs contextes d'utilisation. Cette méthode est généralement entraînée avec un grand corpus de documents. Cette étape permet de créer une matrice M représentant les fréquences des mots dans les documents. Chaque terme est représenté par une ligne et chaque document par une colonne. Cette matrice peut être vue comme un espace vectoriel multidimensionnel où chaque document est représenté par un point dans l'espace des termes.

L'étape clé de *LSA* est l'utilisation de la décomposition en valeurs singulières (*singular value decomposition, SVD*). Afin de réduire le nombre de dimensions de la matrice, cette technique mathématique décompose la matrice en trois facteurs, deux matrices orthogonales U et V, et une matrice diagonale S. Le produit matriciel de ces trois facteurs reconstitue la matrice initiale M. Réduire l'espace initial vers k dimensions revient à mettre à zéro toutes les valeurs de la diagonale de S sauf les k premières. La nouvelle matrice construite M', de rang k, est la plus proche de M au sens des moindres carrés. La *SVD* a pour effet d'éliminer les dimensions les moins significatives, et ainsi de produire un espace sémantiquement plus précis.

Dans nos expériences, on évalue la technique *LSA* pour deux applications, la première est une classification de copies selon le thème, alors que la deuxième est la correction d'essais d'étudiants. Tous les essais sont d'environ deux pages de longueur et se composent de quatre sections correspondant à des réponses à autant de questions. Chaque réponse traite un aspect précis du processus de gestion de projet. Chaque copie de notre corpus est, au préalable, corrigée par deux correcteurs humains.

Pour les expériences de classification, le corpus de travaux se compose d'essais de deux pages portant sur un thème parmi trois. Nous avons testé *LSA* avec des espaces réduits de différentes tailles et nous avons comparé nos résultats avec ceux du modèle d'espace vectoriel (*MEV*). Ce dernier reprend les mêmes étapes que *LSA* sauf qu'il n'utilise pas la réduction de l'espace.

Une première expérience nous a permis de déterminer que les deux méthodes (*LSA* et *MEV*) réussissent une classification correcte des documents selon leurs thèmes. Cette expérience initiale nous a aussi permis d'établir la capacité de discrimination des techniques et d'évaluer l'impact du nombre de dimensions sur l'efficacité de *LSA*.

Nos résultats confirment que *LSA* a un degré de discrimination beaucoup plus prononcé que le *MEV* et que les documents de même thème pouvaient atteindre un niveau de similarité très élevé (0,95 pour *LSA* comparé avec 0,2 pour le *MEV*). Le nombre de dimensions de l'espace réduit à un grand impact sur la qualité des résultats de *LSA*. Ainsi on a remarqué que l'efficacité de *LSA* se dégrade sensiblement avec un très petit ou un grand nombre de dimensions et que les meilleurs résultats résident dans un espace de petite taille relativement à l'espace initial. Dans nos expériences, cette taille est proche de 7% de l'espace initial.

Suite à cette première expérience de classification, nous avons évalué *LSA* pour la tâche plus difficile de correction des copies, à laquelle nous référerons par le terme « *cotation* ». Pour cette expérience, nous disposons de 77 copies que nous divisons en trois groupes. Le premier, appelé « *corpus référence* », contient 26 copies et il est utilisé comme référence pour évaluer le reste des copies. Le deuxième, appelé « *corpus d'entraînement* », comporte 25 copies et il est utilisé pour entraîner et ajuster les paramètres du modèle. Enfin le dernier groupe, appelé « *corpus de validation* », regroupe les 26 copies restantes et on l'utilise pour tester le modèle.

La cotation est approchée selon deux méthodes différentes, la première est une « *cotation modulaire* », cette méthode consiste à diviser la copie en quatre paragraphes, un par question. On corrige chaque question à part et la somme des notes constitue la note générale de la copie.

La deuxième méthode considère la copie entière, toute question confondue, comme un seul texte. Cette méthode compare l'ensemble de la copie avec les textes des copies corrigées. C'est ce qu'on appelle une « *cotation holistique* ».

La cotation, elle-même, peut être réalisée selon plusieurs algorithmes. Nous en avons testé deux. Le premier effectue une cotation selon le principe des « *voisins les plus proches* » c'est-à-dire que chaque copie reçoit une note pondérée des copies corrigées les plus similaires. Le second algorithme effectue une *« cotation par classification »*. À partir des copies corrigées à notre disposition, on construit trois catégories *A, B* et *C* qui sont respectivement, les bonnes, les moyennes et les mauvaises copies. On crée un vecteur moyen par catégorie puis on classifie les copies selon leur degré de similarité avec ces vecteurs, ainsi chaque copie aura une cote *A, B* ou *C* au lieu d'une note précise.

Pour chaque expérience, on compare *LSA* avec le modèle d'espace vectoriel, la comparaison est basée sur le niveau de la corrélation avec les notes attribuées par les correcteurs humains.

Les résultats obtenus confirment que *LSA* effectue une meilleure cotation que le *MEV*. Pour la cotation holistique, nous avons observé que les notes attribuées par *LSA* selon les voisins les plus proches ont un taux de corrélation de 0,60 avec les correcteurs humains contre un taux de 0,37 pour le *MEV*. La classification holistique, testée sur un ensemble de 26 copies, était aussi meilleure avec *LSA* (53%) qu'avec le *MEV* (34%).

Pour la cotation modulaire, les notes par copie attribuées par *LSA* selon les voisins les plus proches ont obtenu un niveau de corrélation de 0,42 contre 0,22 pour le *MEV*. Tandis que, sur l'ensemble des questions, les deux méthodes ont eu des performances presque égales pour la cotation par classification (≈55% sur les 104 réponses).

Pour les deux approches « *holistique* » et « *modulaire*», les expériences ont confirmé que *LSA* est plus performante que le *MEV*. De ce fait, la technique *LSA* atteint un niveau de corrélation avec les correcteurs presque égal à celui d'inter-correcteur (la corrélation inter-correcteur est 0,44 ce qui demeure faible pour ce type de problème).

Notre évaluation de LSA a affirmé le grand potentiel de cette technique. Les résultats de la classification ainsi que la cotation des essais avec *LSA* sont encourageants. La prochaine étape serait d'utiliser cette application pour la correction des études de cas d'une matière enseignée à l'École polytechnique et peut être mettre cet outil à la disposition des étudiants via une interface web.

Abstract

Essay assignment is one of the most valuable tools used to evaluate the student's comprehension of a course. It provides a clear estimation not only about the understanding of the course fundamentals but also the student's skills to express and organize his or her knowledge in a coherent writing.

Although there is no doubt about the essays usefulness and richness in the academic application, it has considerable disadvantages like the time and resources consumed for the grading as well as the subjectivity and expertise involved. The automated grading offers an interesting avenue to overcome these disadvantages.

With the current digital revolution, students usually return a digital copy of their work, and as a result it is now more interesting to consider the automated grading option than ever before. This idea was subject of much research since the early sixties and many attempts were made to achieve the same quality level as the human grading. In most of these automatic grading experiences, the evaluations were based on simple criteria such as the number of words and paragraphs used, the average word length, etc.

The problem with this kind of criteria is that it does not consider the "meaning" of words: the idea behind the words is more important than the words used to express it. Due to the lack of semantic, these methods are facing many problems. One problem is the "*synonymy of words*", which refers to different words referring to the same concept such as "*car*", "*automobile*", or "*vehicle*". A second problem is "*polysemy*", which is the capacity for a word to have multiple meanings like the word "*mole*" that has more than seven different meanings: It can refer to an animal, a spy working under deep cover, or a Mexican sauce.

In this thesis, we evaluate a technique called "*Latent Semantic Analysis*" (*LSA*) for the automatic grading of two page essays in French. *LSA* uses the words frequency statistics over documents to explore the high-level relations between words. This technique relies on the "*Singular Value Decomposition*" (*SVD*) to extract the word's meaning based on its context. The *SVD* reduces the space dimension's number by keeping only the most significant ones, therefore making the new space semantically more precise.

We evaluate *LSA* for two different tasks: document themes classification and document grading. In the document classification task, we use a collection of documents from three different topics; each related to only one of the topics. We use *LSA* to categorise each document in one of the three topics. The results of the classification task for LSA is compared to the "Vector Space Model" (*VSM*) which is similar to the *LSA* but perform dimension reduction.

In our experience, the collection contained 30 documents which can be categorized in three topics each containing 10 documents. Based on the subject, both methods successfully categorised the collection into the three groups.

Although the results show that *VSM* and *LSA* both made correct categorisation, we noticed an important difference in the quality of the results. In this task, we found that the *LSA* has a greater discrimination power than the SVM and therefore the first one classifies more precisely than the other.

After evaluating *LSA* for document classification, we turn our attention to its ability to grade the essays. Using the *LSA* ability to detect the latent semantic relations between copies, we grade new essays by measuring their degree of similarity with pre-graded essays.

In our experiments, we have a collection of 77 essays each containing the answers to four questions assignment submitted by students. All the essays have been graded by two human graders. We divided these essays in three groups. The first group, which contains 26 essays, is used as "*references*" to grade the rest of the copies. The second group, called "*training*", contains 25 essays and is used to calibrate the model. The last group containing 26 essays is called "*validation*" and used to test the method. Since the questions are independent from each other, we decided to approach the grading task from two different ways. First, by grading the whole copy as a merged text which we call the "*Holistic grading*". Second, by grading each question individually and then considering the sum of the grades as the copy's grade, which we call "*Modular grading*".

For each approach, we graded the target copies with two methods. In the first method called "*Nearest neighbour*", the copy's score is a weighted value of its nearest reference copies. In the second method called "*Grading by classification*", the reference copies were categorized in three quality levels named best, medium and low. Then each level is represented by the average of all copies in it. Finally, each copy from the validation category is evaluated against all the levels to find the closest level corresponding to the copy and subsequently assigned its grade.

Using the holistic grading, based on the nearest neighbour copies, the correlation of the *LSA* 's grading with human evaluation is 0.60 versus 0.37 for the VSM. Using the same holistic grading but based on grading by classification, *LSA* was 53% of the time correct versus 34% for the *VSM.*

The second grading method is the modular grading. When we used this method with the nearest neighbour approach, *LSA* grading correlation value to human graders is 0.42 versus 0.22 for the *MEV* method. When grading using the same modular approach but based on grading by classification, both methods were approximately equal and were 55% of the time correct.

The LSA basis is the dimensions reduction which captures the most important information in the matrix and ignores the rest. However, there is no direct method to estimate the optimum number of dimensions to keep in the reduced semantic space. This parameter is a result of multiple tests which can be time consuming in the case of large semantic spaces. The optimum number of dimensions in the semantic space is very important parameter and should be carefully selected.

In our experiments, the latent semantic analysis is considered relatively successful for grading essays for which the semantic dimension is most important. The *LSA* shows high topic discrimination in the categorization tests and good quality sensitivity in the essays grading tests.

Our experiences confirm that *LSA* grading is confirmed in to be as good as the human grading for French essays of short answers. This work maybe considered as the first step to build an automated essays grader for "*École polytechnique*" with a human friendly graphic interface and a better quality corpus.

Table des matières

Liste des tableaux

Liste des figures

Liste des sigles et abréviations

LSA	Analyse sémantique latente *Latent semantic analysis*
MEV	Modèle d'espace vectoriel
RI	Recherche d'information

Introduction

Les évaluations et les examens sont d'une importance primordiale dans le processus d'enseignement car ils permettent d'évaluer à quel point les étudiants ont assimilé le contenu du cours.

Pour les enseignants, c'est une occasion de revoir et de perfectionner leurs méthodes d'enseignement et peut être penser à enrichir le contenu du cours pour l'adapter au niveau des étudiants et ainsi le rendre plus profitable pour ces derniers. Pour les étudiants, les évaluations leur permettent d'identifier leurs lacunes de connaissance et les parties du cours à travailler, sans oublier que c'est une façon de se comparer à leurs camarades.

La dissertation est une des méthodes d'évaluation les plus connues et les plus efficaces. En plus de refléter le niveau de compréhension du cours, la dissertation permet aussi d'apprécier la cohérence des idées et la fluidité d'expression écrite chez les étudiants. Cette méthode riche et pédagogiquement bénéfique pousse les étudiants à réfléchir, à organiser leurs idées et à les exprimer librement, contrairement aux autres méthodes d'évaluation comme le questionnaire à choix multiples.

Malgré ses avantages multiples, l'évaluation par dissertation n'est pas toujours le premier choix des enseignants. Cette méthode consomme beaucoup de temps et dépend entièrement du bon jugement du correcteur. Ce dernier doit juger selon les mêmes critères et être objectif tout au long de la correction. Cette objectivité est souvent remise en question à cause des divergences de notation. D'ailleurs, c'est pour cette raison qu'on applique souvent la double correction dans les examens de recrutement et les compétitions de haut niveau.

La correction automatique des textes est une solution qui assure une correction équitable de toutes les copies, en plus de réduire considérablement le temps nécessaire pour l'accomplissement de cette tâche. Il existe diverses méthodes de correction automatique et plusieurs d'entre elles sont déjà commercialisées. À titre d'exemple on cite : *Project Essay Grade* (PEG), *Intelligent Essay Assessor* (IEA) ou encore *Electronic Essay Rater* (E-Rater).

Les approches sont différentes mais le but est toujours le même : créer un outil efficace et fiable qui peut évaluer un texte aussi bien qu'un humain. L'évaluation humaine des copies est un processus complexe combinant un ensemble de critères sémantiques, syntaxiques et qualitatifs. Les approches actuelles ont fait de grands progrès pour simuler ce processus et l'idée de la correction automatique gagne du terrain et sera dans un avenir proche un outil d'enseignement indispensable aussi bien pour les professeurs que pour les étudiants.

Dans ce mémoire, il est question d'évaluer la technique d'analyse sémantique latente (*Latent Semantic Analysis*, désormais *LSA*) pour la correction automatique des textes. Cette technique permet une correction selon le contexe des mots. Elle juge le contenu des copies par une approche mathématique qui vise à approcher le jugement d'un correcteur humain.

Notre objectif est d'implémenter et évaluer la performance de *LSA* pour une correction de textes en français de deux pages et de comparer cette méthode avec un modèle similaire mais plus simple, à savoir, le modèle d'espace vectoriel (désormais *MEV*). La différence majeure entre ces deux modèles est que, contrairement au *MEV*, *LSA* utilise la décomposition en valeurs singulières (*Singular Value Decomposition* désormais *SVD*). Cette technique mathématique matricielle permet à *LSA* d'extraire les relations latentes entre les termes grâce aux associations d'ordre supérieur restées jusqu'à maintenant inexploitables par le *MEV*.

Les approches *MEV* et *LSA* sont issues du domaine de la recherche d'information (*Information Retrieval*, désormais *RI*). Le premier chapitre est donc dédié à ce domaine et présente les problématiques générales et sous-jacentes aux techniques étudiées. On y présente le domaine, des exemples de ses modèles et quelques applications.

Nous présentons dans le deuxième chapitre le modèle d'espace vectoriel qui est à la base de l'approche *LSA*. Nous discutons des fondements mathématiques de ce modèle, aussi bien que de ses expériences et ses limites.

Le troisième chapitre est consacré à la technique *LSA*. On discute de son apport, de son fonctionnement et surtout de l'utilisation de la décomposition en valeurs singulières (*SVD*).

Dans le chapitre quatre, nous détaillons les deux applications de *LSA*, d'abord la classification de textes puis la cotation des copies. Nous présentons l'implémentation du *LSA* pour chaque application avec les résultats des tests, ensuite on compare ces derniers avec ceux du *MEV* et aussi avec les résultats de la cotation humaine.

Enfin, la conclusion est consacrée aux limitations et aux travaux futurs qui peuvent étendre ce travail et améliorer la cotation automatique des analyses de cas.

Chapitre 1 : Modèles de recherche d'information

La cotation automatique des essais des étudiants emprunte plusieurs techniques du domaine de la recherche d'information (*RI*). Grâce à la révolution numérique actuelle, le domaine de la *RI* a suscité beaucoup d'intérêt. L'abondance de l'information numérique a mis en évidence l'importance des algorithmes de recherche. Parcourir tous les documents à chaque recherche est devenu impraticable et il fallait remplacer ce genre de recherche par des stratégies efficaces qui donnent un accès rapide (direct ou rediriger) à l'information pertinente.

Les bases de la cotation automatique sont héritées du domaine de la RI, par exemple la représentation des copies, les méthodes de pondération et le calcul de similarité. Cependant, la cotation a aussi hérité les problèmes de la RI comme la polysémie et la synonymie. Pour bien cerner les principes de la RI, il faut comprendre ses modèles de recherche les plus connus à savoir le modèle booléen, le modèle probabiliste et le modèle d'espace vectoriel. La compréhension du fonctionnement de ces modèles permettra une meilleure maîtrise de *LSA*, en plus d'offrir un modèle de référence (*MEV*) pour évaluer la qualité des résultats obtenus.

1.1 Introduction

La recherche d'information date des années quarante. Cette branche de recherche existe depuis l'invention de l'ordinateur et elle continuera à évoluer et à gagner de l'intérêt avec l'expansion de l'utilisation des documents numériques et la vulgarisation de l'informatique. Une définition claire et précise de la *RI* est celle de Mooers (1958) :

> *"Information retrieval is the name of the process or method whereby a prospective user of information is able to convert his need for information into an actual list of citations to documents in storage containing information useful to him."*

La *RI* a pour but de trouver une information précise dans une grande collection de document et de rediriger l'utilisateur vers les documents qui répondent à ses besoins. Les systèmes de RI jouent le rôle d'interface entre l'utilisateur et la collection de documents. Ces systèmes maintiennent une représentation interne des documents de la collection appelée « *index* » (Plus de détails dans le chapitre 2 « Modèle d'espace vectoriel », Étape 4 « Fichier index »). En interrogeant cette représentation (interne, relativement petite et bien organisée) on économise considérablement, en termes de temps de réponse et des ressources systèmes utilisées.

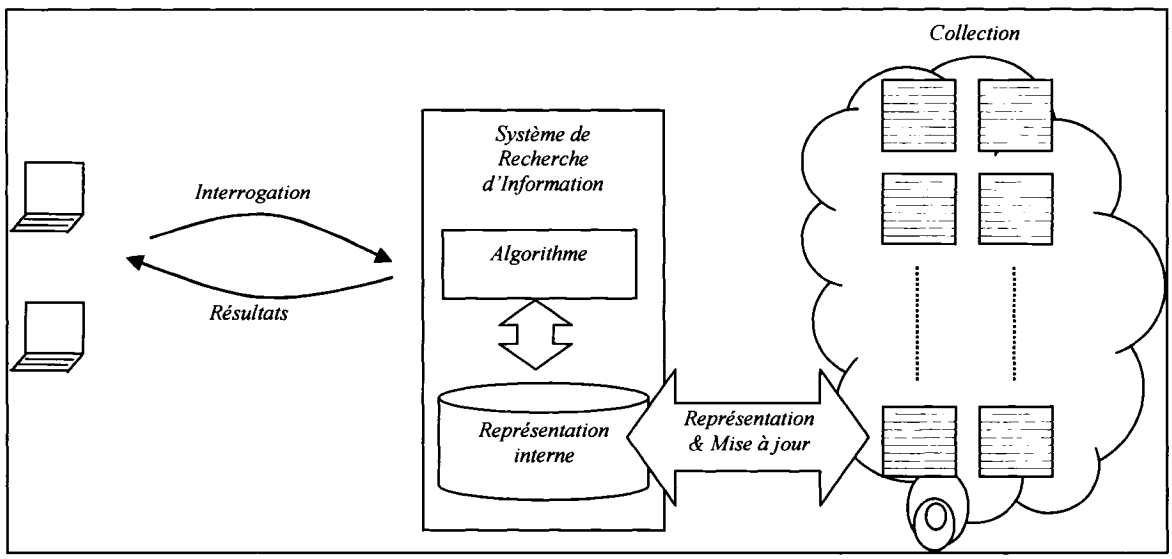

Figure 1.1 : Fonctionnement d'un système de recherche d'information

1.2 Notions de « précision » et « rappel »

Afin d'évaluer la qualité d'une recherche – la pertinence des résultats retournés – deux critères principaux sont utilisés : la *précision* et le *rappel*.

Supposons qu'on a une collection de 100 documents dont 20 documents sont pertinents. Si une recherche retourne 10 résultats dont 5 documents sont pertinents, alors on a une précision de 50% (5 pertinents sur les 10 résultats retournés) et un rappel de 25% (5 pertinents sur les 20 qui existent dans la collection).

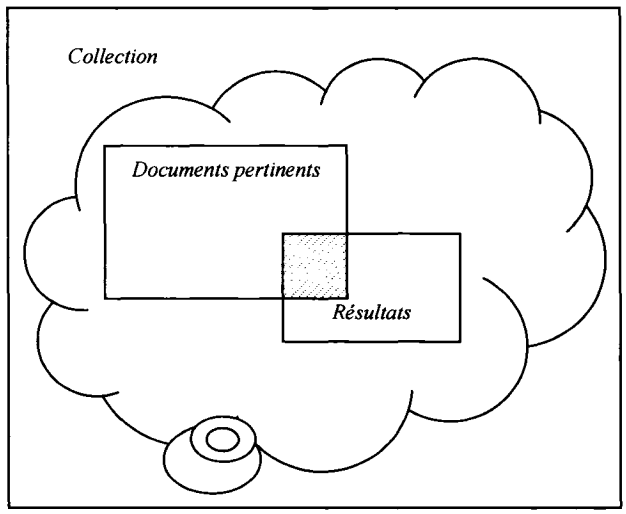

Figure 1.2 : exemple de résultats de recherche

Le rappel est calculé par rapport aux documents pertinents dans toute la collection, dans le meilleur des cas, on aura un rappel de 100% ce qui veut dire que tous les documents pertinents sont retournés.

La précision est calculée par rapport aux résultats, une précision parfaite signifie que tous les documents retournés sont pertinents pour la recherche.

Une méthode de recherche sera parfaite si elle retourne tous, et rien que, les documents pertinents (100% de rappel et 100% de précision). Malheureusement une technique aussi performante n'existe pas pour deux raisons. Premièrement, cela suppose qu'on peut recenser tous les documents pertinents pour chaque recherche, ce qui est pratiquement impossible. La deuxième raison est que la pertinence est un critère subjectif qui peut varier selon le bon jugement de l'utilisateur.

Dans les sections suivantes, nous présentons les modèles les plus connus de la *RI*, en commençant par le plus ancien, à savoir le modèle booléen.

1.3 Modèle Booléen

Le modèle booléen est le modèle le plus simple et le plus ancien, il se base sur la recherche des mots de la requête dans une collection de documents. La requête est définie comme une expression logique qui détermine une combinaison de présence et d'absence des mots dans les documents. Ce modèle de recherche est qualifié de *strict* car il ne considère que les documents qui vérifient toutes les conditions de la requête.

Un grand avantage de ce modèle est l'utilisation des opérateurs logiques *ET*, *OU* et *NON* pour exprimer les requêtes de recherche, ce qui offre plus de souplesse et une meilleure description de l'information recherchée.

Pour illustrer le fonctionnement de ce modèle, supposons qu'on a une petite collection de trois documents {*Doc1, Doc2, Doc3*} et que notre requête, *Req*, se compose de quatre termes *T1*, *T2*, *T3* et *T4* :

Req = (T1 **OU** *T2)* **ET** *((***NON** *T3)* **ET** *T4)*

La distribution des termes dans les documents de la collection est une matrice *M* où les documents sont représentés par les colonnes et les termes par les lignes :

Tableau 1.1 : Fréquences des mots pour un modèle booléen

	Doc1	Doc2	Doc3
T1	0	1	1
T2	1	0	1
T3	0	1	0
T4	1	0	0

Une cellule de valeur 0 signifie que le mot est absent, et une valeur de 1 veut dire que le terme apparaît dans le document.

Notre recherche peut être divisée en deux sous-recherches, *Req1* et *Req2* :

Req1 : T1 **OU** *T2*

*Req2 : (***NON** *T3)* **ET** *T4*

La première recherche est une union ou sommation (**OU**) des ensembles qui citent les termes *T1* ou *T2*, alors que la deuxième recherche est une intersection ou une soustraction (**ET**), c'est-à-dire qu'on retire de l'ensemble des documents qui citent le terme *T4* tous ceux qui citent le terme *T3*.

Les résultats des sous-recherches sont :

$Req1 \leftrightarrow (M[Doc_i, T1]=1)$ **OU** $(M[Doc_i, T2]=1)$ où $i = 1..3$

$Req1 = \{Doc1, Doc2, Doc3\}$

$Req2 \leftrightarrow (M[Doc_i, T3]=0)$ **ET** $(M[Doc_i, T4]=1)$ où $i = 1..3$

$Req2 = \{Doc1\}$

La réponse à notre recherche *Req* est l'intersection de l'ensemble des résultats de *Req1* avec celui de *Req2*, dans ce cas :

Req = *Req1* **ET** *Req2*

Résultat = $\{Doc1\}$

Dans l'exemple, on a un seul document qui répond à 100% à la recherche. Ce modèle adopte une approche stricte car tous les autres documents sont considérés non pertinents même s'ils répondent partiellement à la recherche. Une autre remarque est que ce modèle n'offre pas de critère pour trier les résultats par ordre de pertinence.

Le fonctionnement du modèle booléen est simple mais il est loin d'être parfait, Salton [31] a résumé les points faibles du modèle booléen :

- L'utilisateur n'a aucun contrôle sur le nombre des résultats retournés : un document est pertinent si et seulement s'il répond à tous les critères de la

recherche. Dans ce cas, peu de critères retournent trop de résultats et trop de critères retournent très peu sinon aucun résultat. L'utilisateur doit, donc, trouver lui-même le juste milieu entre le nombre de critère et la taille des résultats et ainsi re-formuler sa recherche jusqu'à 'satisfaction'.

- Tous les termes, de la requête et des documents, ont le même poids : pour une requête de type : « *T1* **OU** *T2* », il n'y a aucun moyen de comparer la pertinence d'un document qui cite seulement le terme *T1* avec un autre qui cite *T2*. Tous les termes et tous les résultats sont égaux en importance.
- Tous les documents retournés ont le même degré de pertinence : toutes les cellules du tableau 1.1 ont une valeur de 0 si le terme n'apparaît pas dans le document ou 1 s'il apparaît une ou plusieurs fois. Cette représentation affaiblit le niveau de distinction entre documents et élimine la possibilité d'utiliser le nombre d'occurrence comme mesure de pertinence.
- La logique booléenne ignore la pertinence partielle : Pour une requête « **OU** » avec plusieurs conditions, un document qui satisfait toutes les conditions est considéré aussi important qu'un autre qui vérifie une seule condition. De la même manière, pour une requête « **ET** » avec plusieurs conditions, un document qui satisfait toutes les conditions à l'exception d'une seule est considéré étant aussi inutile qu'un autre qui ne répond à aucune des conditions. Ainsi la pertinence partielle est complètement masquée.

Le point fort du modèle booléen est qu'il donne une grande liberté pour la formulation des requêtes, ce qui est très utile pour exprimer des recherches simples. Cependant, cette tâche n'est pas aussi plaisante pour des requêtes complexes, surtout pour les utilisateurs non expérimentés qui ne maîtrisent pas bien la logique des opérateurs booléens.

Une amélioration importante de ce modèle consiste à reformuler la requête. Dans une recherche avec un nombre important de conditions, il est fort possible que le système ne retournera pas de résultat. De ce fait, une reformulation de la recherche en éliminant quelques critères serait souhaitable même si les résultats seront des documents partiellement pertinents. Le modèle booléen étendu apporte plusieurs améliorations surtout au niveau de la pondération et le tri des résultats.

1.4 Modèle booléen étendu

On reproche au modèle booléen classique l'absence de pondération des termes et pour les requêtes et pour les documents, ainsi que son évaluation stricte des documents (toutes les conditions doivent être vérifiées). La version étendue introduit de nouveaux paramètres pour améliorer l'efficacité du modèle booléen "classique".

Les deux opérateurs booléens « ET » et « OU » permettent d'exprimer deux requêtes de base « *T1* **ET** *T2* » et « *T1* **OU** *T2* », où *T1* et *T2* sont deux termes. Cela revient à classer les documents en trois catégories : ceux qui citent *T1* et *T2*, ceux qui citent soit *T1* soit *T2* et ceux qui citent aucun des deux termes. Le tableau 1.2 représente les quatre cas de figure possibles avec une recherche de deux termes et leurs poids dans un modèle booléen classique.

Tableau 1.2 : Résultats d'une requête de base

	Doc1	Doc2	Doc3	Doc4
T1	1	1	0	0
T2	1	0	1	0
T1 *OU* T2	1	1	1	0
T1 *ET* T2	1	0	0	0

Salton [31] représente les documents dans un espace à deux dimensions (une dimension par terme de requête), ainsi pour que la requête « *T1* ***ET*** *T2* » soit vérifiée, égale à 1, il faut que les deux termes soient présents ce qui est représenté par le point (1,1), et pour que « *T1* ***OU*** *T2* » soit vérifiée, égale à 1, il suffit d'éviter l'absence des deux termes, c'est-à-dire éviter le point (0,0). De cette façon, on peut calculer la similarité comme une distance entre deux points. Les documents les plus pertinents pour la requête « *T1* ***ET*** *T2* » sont les plus proches du point (1,1) (la similarité est inversement proportionnelle à cette distance), et les plus pertinents pour « *T1* ***OU*** *T2* » sont les plus éloignés du point (0,0) (la similarité est proportionnelle à cette distance), figure 1.3.

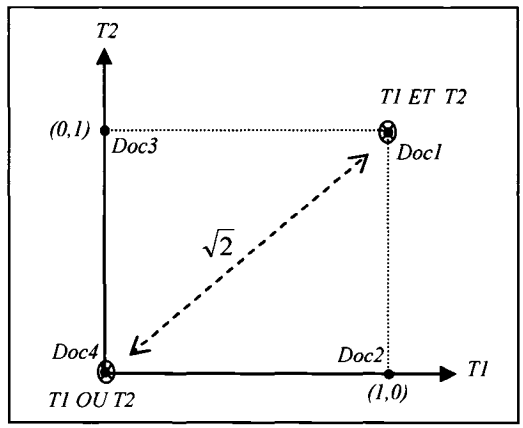

Figure 1.3 : Concept de similarité avec les requêtes de base

Par exemple, la distance entre un document $Doc(x,y)$ et le point (1,1) est calculée par la formule : $\sqrt{(1-x)^2+(1-y)^2}$ et celle qui le sépare du point (0,0) par la formule $\sqrt{x^2+y^2}$. Salton [31] normalise les valeurs en divisant par $\sqrt{2}$. Cette valeur correspond à la distance entre les deux points caractéristiques des deux requêtes de base (distance maximale).

On définit les deux fonction de similarité, Sim_{ET} et Sim_{OU}, pour le modèle étendu :

$$Sim_{OU} = \sqrt{\frac{x^2 + y^2}{2}}$$

$$Sim_{ET} = 1 - \sqrt{\frac{(1-x)^2 + (1-y)^2}{2}}$$

Le tableau de similarité avec les nouvelles formules de similarité :

Tableau 1.3 : Valeurs de similarité avec les requêtes de base

	Doc1	Doc2	Doc3	Doc4
T1	1	1	0	0
T2	1	0	1	0
T1 *ET* T2	1	$1/\sqrt{2}$	$1/\sqrt{2}$	0
T1 *OU* T2	1	$1-1/\sqrt{2}$	$1-1/\sqrt{2}$	0

Cette méthode de calcul permet de rendre le modèle plus flexible, la similarité n'est plus binaire mais une valeur réelle qui varie entre [0,1]. Ainsi pour une requête « *T1* **ET** *T2* », les documents qui citent juste un de ces deux termes (*Doc2* ou *Doc3*) sont mieux classés qu'un document qui cite aucun des deux termes (*Doc4*). De la même façon, pour une requête « *T1* **OU** *T2* », un document qui cite les deux termes (*Doc1*) est mieux classé que les documents qui citent juste un seul terme (*Doc2* ou *Doc3*).

On reproche au modèle booléen classique qu'il considère tous les termes d'importance égale aussi bien dans les documents que dans les requêtes. Le modèle étendu propose d'affecter à chaque terme un poids associé entre 0 et 1 reflétant son importance et dans la requête et dans le document. La pondération des termes de la requête est donc à la charge de l'utilisateur, ce dernier doit les adapter selon ses besoins. Ainsi l'utilisateur doit d'abord bien sélectionner les termes, bien comprendre la logique des opérateurs booléens et en plus pondérer de manière précise les termes pour optimiser les résultats. Cette tâche n'est pas évidente pour des utilisateurs moyens et limite l'utilisation de ce modèle par le grand public.

1.5 Modèle probabiliste

Dans les modèles booléens présentés, il est toujours question de déterminer si un document est pertinent ou non, le modèle probabiliste adopte une approche différente : on cherche à évaluer la probabilité qu'un document soit pertinent pour une recherche précise. Dans ce qui suit on suppose qu'on répond à une seule requête notée *Rq*.

Le modèle probabiliste date des années soixante. La plus ancienne référence qu'on a pu trouver est celle de Maron et Khurn datant de 1960 et intitulée : « *On relevance, probabilistic indexing and information retrieval*» [18]. Dans cet article, ils décrivent une technique « innovatrice » qui reprend le concept de « la pertinence » avec une théorie de calcul de probabilité. Le but est que l'ordinateur puisse répondre à une recherche avec une liste de documents triés selon leur probabilité de pertinence.

Dans ce qui suit, nous nous basons sur les articles de Rijsbergen [25] et Greengrass [9] pour expliquer les bases du modèle probabiliste.

Pour un document *D*, la probabilité qu'il soit pertinent (*relevant*) est représentée par $P(R|D)$ et la probabilité qu'il soit non pertinent par $P(NR|D)$. Avant de présenter les formules, nous allons introduire l'hypothèse de base du modèle probabiliste : la règle de *« indépendance binaire »*. Ce modèle suppose que la présence ou l'absence d'un terme dans un document est complètement indépendante de celle de tout autre terme. Aussi la pertinence ou la non-pertinence d'un document est indépendante de celle de tout autre document de la collection. Le résultat direct de cette hypothèse est la formule de calcul des probabilités : la probabilité de deux événements est égale au produit de la probabilité de chacun d'eux, formule 1.1, et la même chose pour la pertinence (formule 1.2) et la non pertinence (formule 1.3) :

(Formule 1.1) $$P(A,B) = P(A)\,P(B)$$

(Formule 1.2) $$P(A,B|R) = P(A|R)\,P(B|R)$$

(Formule 1.3) $P(A,B|NR) = P(A|NR)\,P(B|NR)$

Les valeurs de $P(R|D)$ et $P(NR|D)$ nous permettront de trier les documents trouvés selon l'ordre de probabilité de leurs pertinences, ainsi on définit une fonction *Ordre* :

(Formule 1.4) $$Ordre(D) = \frac{P(R \mid D)}{P(NR \mid D)}$$

$P(R|D)$ est la probabilité qu'un document trouvé soit pertinent. Il n'y a pas de façon directe pour calculer cette probabilité. Pour cette raison, on utilise le théorème de Bayes pour exprimer cette dernière en fonction de $P(D|R)$, la probabilité de choisir le document D sur l'ensemble pertinent.

$$P(R \mid D) = \frac{P(D \mid R)P(R)}{P(D)} \quad \text{Et} \quad P(NR \mid D) = \frac{P(D \mid NR)P(NR)}{P(D)}$$

$P(R)$ et $P(NR)$ sont respectivement la probabilité de pertinence et de la non pertinence sur toute la collection. Pour la même requête Rq, ces deux probabilités sont constantes. La fonction *Ordre* est donc proportionnelle au ratio $P(D|R) / P(D|NR)$ et ainsi on peut la redéfinir comme étant :

(Formule 1.5) $$Ordre(D) \propto \frac{P(D \mid R)}{P(D \mid NR)}$$

L'index recense l'ensemble des termes de la collection, alors chaque document peut être représenté par un vecteur de termes où chaque ordonnée a une valeur de 0 ou 1 selon l'absence ou la présence du terme dans le document.

$$D = D(t_1 = x_1, t_2 = x_2, t_3 = x_3, ..., t_n = x_n) \text{ où } x_i \in \{0,1\}$$

À ce point, on peut juger de la pertinence d'un document selon la pertinence de ses termes :

$$P(D \mid R) = P(t_1 = x_1, t_2 = x_2, t_3 = x_3, ..., t_n = x_n \mid R) \text{ où } x_i \in \{0,1\}$$

Selon le principe d'indépendance :

$$P(D \mid R) = P(t_1 = x_1 \mid R) * P(t_2 = x_2 \mid R) * .... * P(t_n = x_n \mid R)$$

$$P(D \mid R) = \prod_i^n P(t_i = x_i \mid R)$$

Et puisque $P(t_i = 1 \mid R) + P(t_i = 0 \mid R) = 1$

Alors $$P(D \mid R) = \prod_i^n P(t_i = 1 \mid R)^{x_i} * P(t_i = 0 \mid R)^{1-x_i}$$

où $x_i \in \{0,1\}$

Le problème est transformé en un calcul des probabilités de pertinence des termes du document. Pour estimer ces probabilités, on considère qu'on dispose d'un échantillon représentatif de la collection contenant N documents pré-classifiés dont n documents citent le terme t. on suppose que C documents sont pertinents à notre requête Rq, tableau 1.4.

On considère deux variables : $P_R = P(t = 1 \mid R)$ la probabilité que le terme t apparaît dans un document pertinent et $P_{NR} = P(t = 1 \mid NR)$ la probabilité que t apparaît dans un document non pertinent.

Tableau 1.4 : Distribution d'un terme* t *dans les documents de l'échantillon

	D est pertinent	*D* n'est pas pertinent	Total
D cite *t*	*c*	*n - c*	*n*
D ne cite pas *t*	*C – c*	*(N – C) – (n – c)*	*N - n*

Total	C	$N-C$	N

On réécrit les valeurs des probabilités en fonction de ces variables :

$$P(t_i = 1 \mid R) = P_{Ri} \quad \Rightarrow \quad P_R = \frac{c}{C}$$

$$P(t_i = 0 \mid R) = 1 - P_{Ri} \quad \Rightarrow \quad 1 - P_R = \frac{C-c}{C}$$

$$P(t_i = 1 \mid NR) = P_{NRi} \quad \Rightarrow \quad P_{NR} = \frac{n-c}{N-C}$$

$$P(t_i = 0 \mid NR) = 1 - P_{NRi} \quad \Rightarrow \quad 1 - P_{NR} = \frac{(N-C)-(n-c)}{N-C}$$

On remplace les probabilités dans la fonction *ordre,* la formule 1.5 :

$$Ordre(D) = \frac{\prod_i^n P_R^{xi}(1-P_R)^{1-x_i}}{\prod_i^n P_{NR}^{xi}(1-P_{NR})^{1-x_i}}$$

On considère la fonction G tel que :

$$G\,(D) = log\,(Ordre\,(D))$$

$$G(D) = \log(\prod_i^n P_R^{xi}(1-P_R)^{1-x_i}) - \log(\prod_i^n P_{NR}^{xi}(1-P_{NR})^{1-x_i})$$

$$G(D) = \sum_i^n \log(P_R^{xi}(1-P_R)^{1-x_i}) - \sum_i^n \log(P_{NR}^{xi}(1-P_{NR})^{1-x_i})$$

$$G(D)=\sum_{i=1}^{n}(log(P_R^{xi})+log(1-P_R)^{1-xi})-\sum_{i=1}^{n}(log(P_{NR}^{xi})+log(1-P_{NR})^{1-xi})$$

$$G(D)=\sum_{i=1}^{n}(x_i\,log(P_R)+(1-x_i)log(1-P_R))-\sum_{i=1}^{n}(x_i\,log(P_{NR})+(1-x_i)log(1-P_{NR}))$$

$$G(D)=\sum_{i=1}^{n}x_i(log(\frac{P_R}{1-P_R})+log(1-P_R))-\sum_{i=1}^{n}x_i(log(\frac{P_{NR}}{1-P_R})+log(1-P_{NR}))$$

$$G(D)=\sum_{i}^{n}x_i[\log(\frac{P_R}{1-P_R})-\log(\frac{P_{NR}}{1-P_{NR}})]+\sum_{i}^{n}\log(\frac{1-P_R}{1-P_{NR}})$$

$$G(D)=\sum_{i=1}^{n}x_i\log\frac{P_R(1-P_{NR})}{P_{NR}(1-P_R)}+Cte$$

Où *Cte* est une constante de valeur :

$$Cte=\sum_{i=1}^{n}\log\frac{1-P_R}{1-P_{NR}}$$

Pour simplifier la formule on met :

$$w_i=log\frac{P_R(1-P_{NR})}{P_{NR}(1-P_R)}$$

Finalement, la probabilité d'un document *D* est évaluée en fonction des probabilités de ses termes, c'est-à-dire les termes dont le $x_i=1$:

$$G(D)=\sum_{i=1}^{n}x_iw_i+Cte$$

L'avantage principal du modèle probabiliste est qu'il est basé sur une méthode mathématique prouvée. Cependant, pour adapter la réalité à cette logique, il faut accepter des hypothèses de base qui ne sont pas correctes à 100%, notamment le principe d'indépendance binaire. En réalité deux termes sémantiquement liés ont plus tendance à apparaître ou pas dans le même contexte comme les termes « *clavier* » et « *souris* » ou les termes « *docteur* » et « *hôpital* ».

Cooper [5] a révisé le postulat d'indépendance binaire et il a trouvé que la combinaison des règles d'indépendance peut mener à des résultats erronées.

Il prend l'exemple d'un sous-ensemble, tiré d'une collection, classé selon deux critères A et B tel que :

$$\begin{cases} P(A) = P(B) = 0{,}1 \\ P(R) = 0{,}1 \\ P(R \mid A) = 0{,}5 \\ P(R \mid B) = 0{,}5 \end{cases}$$

Pour calculer la probabilité de $P(A,B)$, on utilise le premier principe d'indépendance, formule 1.1:

$$\begin{aligned} P2 &= P(A,B) \\ &= P(A) * P(B) \\ &= 0.01 \end{aligned}$$

Rappelons ici que selon le principe de l'indépendance de la pertinence :

$$P(A,B \mid R) = P(A \mid R) * P(B \mid R)$$

On calcule la probabilité de trois conditions $P(A,B,R)$ en utilisant une combinaison de ces deux principes :

$$
\begin{aligned}
P3 &= P(A,B,R) \\
&= P(A,B \mid R) * P(R) \\
&= P(A \mid R) * P(B \mid R) * P(R) \\
&= P(R \mid A) * P(A) * P(R \mid B) * P(B) / P(R) \\
&= 0.025
\end{aligned}
$$

On trouve que la probabilité des deux conditions A et B s'améliore en ajoutant une troisième condition R, ce qui est mathématiquement erroné !

Malgré que ces incohérences soient soulignées dans plusieurs articles [26][28][27], les principes d'indépendance sont adoptés pour la simplification des calculs. Cooper [5] a avancé une explication plausible qui justifierait le bon rendement du modèle malgré ces incohérences.

Le modèle probabiliste se base sur un échantillon représentatif pré-classifié pour chaque requête ou plutôt chaque terme de la requête. Ce type échantillon et statistiques ne sont pas toujours disponibles, en plus on ne peut pas garantir que les proportions de pertinence dans l'échantillon soient représentatives de l'ensemble de la collection.

Dans le chapitre suivant, nous présentons un autre type de modèle qui ne nécessite pas de pré-classification. Il s'agit du modèle d'espace vectoriel. Ce dernier adopte une approche différente en se basant sur le calcul des fréquences des mots et la distribution des termes au lieu des probabilités.

Chapitre 2 : Modèle d'espace vectoriel

2.1 Introduction

Le modèle d'espace vectoriel (*MEV*) est dû à Salton [30] en 1975. Il voulait mettre en place une méthode efficace de recherche d'information dans les grandes collections de documents en utilisant des techniques statistiques programmables. On représente la requête ainsi que l'ensemble des documents sous forme de vecteurs, puis on mesure leurs similarités selon la distance qui sépare leurs vecteurs. Le résultat est une liste de documents ordonnée selon le degré de leur « pertinence » par rapport à la requête. La figure 2.1 présente le fonctionnement général du modèle *MEV*.

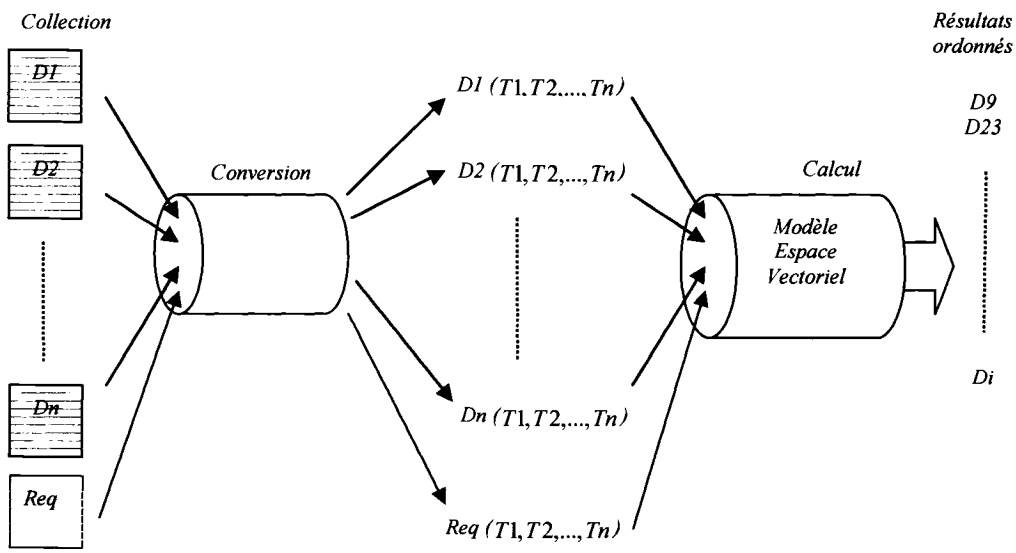

Figure 2.1 : Fonctionnement du modèle d'espace vectoriel

2.2 Étapes du MEV

Avant de présenter le fonctionnement du *MEV*, nous rappelons les notions de base de la RI, par la suite, nous discuterons de l'utilité de chaque étape et de ses techniques d'implantation dans le *MEV*.

2.2.1 Représentation des documents

Avant d'exploiter les documents, il faut d'abord les adapter et formater leurs contenus pour les rendre utilisables par une machine. Par exemple, saisir les documents sur papier puis définir les parties de textes représentatives du contenu soit un résumé ou une liste de mots clés. Cette procédure peut se faire manuellement ou automatiquement par des logiciels de traitement de la langue.

Le but de cette étape est que chaque document de la collection soit représenté par un ensemble de mots qui décrivent son contenu le plus fidèlement possible.

2.2.2 Recherche de la racine

La base des modèles statistiques est le calcul des fréquences des mots dans les documents. Cependant, les variantes du même mot peuvent être utilisées de plusieurs façons différentes dans la même collection. Ces variantes gardent le même sens par exemple les mots « grands », « grandement » et « grandes ». La même chose s'applique pour les formes conjuguées des verbes, ainsi les mots « fait », « faisons », « ferai » et « firent » sont des variantes du verbe « faire ». On suppose que les variantes d'un mot sont sémantiquement équivalentes.

Le but de la recherche de la racine (*stemming*) est d'écrire tous les mots en un format standard pour donner plus d'importance au sens plutôt qu'à la syntaxe. Cette étape permet aussi de diminuer la quantité de mots à comparer et ainsi mettre en évidence les points communs entre les documents.

Il existe deux façons de procéder. La première consiste à chercher une racine commune en supprimant les préfixes et les suffixes les plus longs selon des règles bien précises, c'est ce qu'on appelle la *troncation*. Cette méthode est utilisée depuis le début de la recherche d'information dans les années 1960. Les algorithmes de troncation les plus populaires sont bien décrit par Lovins [15] et Andrews [2].

La deuxième méthode consiste à trouver le mot d'origine (singulier masculin, verbe à l'infinitif) pour chaque terme utilisé en se basant sur un dictionnaire de langue et des analyseurs linguistiques, c'est la *lemmatisation.*

La lemmatisation est plus difficile à implémenter que la troncation. D'ailleurs des expériences ont prouvé que l'apport de la lemmatisation est insignifiant et ne justifie pas son utilisation dans la *RI* en anglais [7], [20] et [11]. Cette conclusion est peut être vraie pour l'anglais mais on ne peut pas l'extrapoler à toutes les langues, surtout les langues morphologiquement plus riches telles que le français, l'italien ou l'hébreu. Il est effectivement prouvé que la lemmatisation est plus performante que la troncation en irlandais [10] et en slovène [22].

2.2.3 Filtrer les « Mots vides »

Une fois qu'on a une description de chaque document, on élimine un ensemble de mots sémantiquement non porteurs d'information. Les mots comme « le, les, la, de, un, une, cet, aussi... » sont largement utilisés dans tous les domaines et peuvent être ignorés sans pour autant affecter la qualité de la recherche. C'est ce qu'on appelle les « Mots vides » (*Stop list*). En supprimant ces mots on réduit sensiblement la taille des descriptions des documents.

Dans chaque domaine, il existe des mots spécifiques et largement utilisés dans les documents relatifs à ce domaine. Ce genre de mots n'est pas un bon critère de recherche. Par exemple, on ne devrait pas rechercher avec le mot « ordinateur » dans une collection dédiée au domaine d'informatique, car cela retournerait la majorité des documents, voire toute la collection. Il faut donc ajouter ce genre de terme à la liste des mots vides et ainsi l'adapter et l'entretenir pour chaque type et thème de collection.

Luhn [17] propose une technique pour remplacer la *stop list*. Partant du principe que la fréquence d'un mot reflète son poids dans la collection, Luhn définit deux seuils de fréquences pour éliminer les mots « bruits », figure 2.2. D'abord il représente l'ensemble des mots de la collection en ordre décroissant de leurs fréquences puis il définit un seuil supérieur qui élimine les mots très fréquents et un seuil inférieur qui supprime les mots rares. Dans les deux cas les mots retirés ne distinguent pas le contenu du document par rapport à la collection. Dans le même sens, Luhn[17] a remarqué que les mots moyennement fréquents reflètent mieux le contenu du document, ainsi l'importance des mots, représentée par la courbe pointillée dans la figure 2.2, atteint son maximum au milieu de l'intervalle délimité par les deux seuils.

Avec cette technique d'élimination, on ne retient que les mots les plus significatifs. Ces derniers sont représentés par l'espace hachuré sur la figure 2.2.

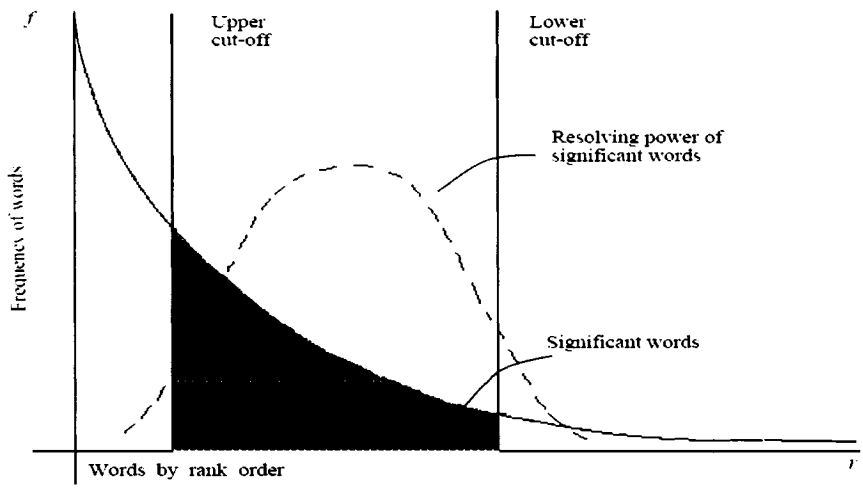

Figure 2.2 : « Stop List » avec les fréquences

Il n'existe pas de méthode directe pour fixer les valeurs des seuils. Ils sont donc obtenus par des tests sur un grand échantillon de documents. Ces valeurs peuvent aussi être ajustées pour améliorer la précision des recherches.

Le but de cette étape est d'éliminer les mots de faibles poids et ainsi améliorer la qualité des recherches et réduire la taille de la représentation interne des documents.

2.2.4 Fichier index

Une procédure simple pour répondre à une requête consiste à chercher les termes de la requête dans chaque document de la collection. Dans le cas de grandes collections, cette technique est excessivement lente. Pour cette raison les systèmes de RI maintiennent une représentation interne résumée, rapide et facile à exploiter décrivant le contenu de chaque document. C'est ce qu'on appelle les fichiers « *index* ».

Les fichiers index sont souvent *inversés*, le principe est le même que celui des pages d'index à la fin des livres. On enregistre, pour chaque terme, l'ensemble des documents qui le citent, le nombre d'occurrences dans chaque document et parfois la position de chaque occurrence. La structure de ces fichiers dépend des besoins de l'application. Par exemple, si on cherche des phrases précises ou des suites de mots dans les titres il serait nécessaire d'enregistrer la position et la hiérarchie de chaque mot dans le document (titre, sous titre, etc.).

Le but de cette étape est d'optimiser la représentation des documents de façon à faciliter leur utilisation et améliorer la rapidité de la recherche.

2.2.5 Pondération des termes

La pondération des termes consiste à affecter un poids (degré d'importance) à chaque terme de la collection. C'est une étape importante dans la recherche d'information car elle définit la manière d'interpréter la fréquence, la présence ou l'absence d'un terme dans le document.

Une pondération simple est d'affecter un poids binaire au terme, soit 1 soit 0 selon que le terme apparaît ou non dans le document. Une version plus évoluée représente l'ensemble des fréquences des termes dans une matrice $M(n \times p)$ avec n le nombre des termes et p le nombre des documents de la collection. La cellule *M [i, j]* a une valeur égale au nombre d'occurrence du terme i dans le document j. Cette représentation est plus significative que les valeurs binaires : le nombre d'occurrences d'un terme nous informe sur son importance dans le document, en plus de mettre en évidence les liaisons éventuelles entre les termes, on peut remarquer, par exemple, si ces derniers sont répétés de manière proportionnelle, inversement proportionnelle ou au hasard.

Le *MEV* se base sur le nombre d'occurrences pour pondérer les mots en utilisant, entre autres, les paramètres de base présentés dans le tableau 2.1.

Tableau 2.1 : Paramètres de pondération

Paramètre	Symbole	Signification
Fréquence du terme	$tf_{i,j}$	Le nombre des occurrences du terme i dans le document j
Fréquence dans les documents	df_i	Le nombre de documents qui citent le terme i
Fréquence dans la collection	cf_i	Le nombre total des occurrences du terme i dans la collection

Il est à noter que : $\sum_j tf_{i,j} = cf_i$ et $df_i \leq cf_i$

La fréquence du terme « tf » donne une idée de son poids au sein d'un document et à quel point ce terme reflète le contenu de ce document. Cependant, la valeur brute de tf n'est pas efficace pour comparer les documents. Par exemple, si un document D_1 cite le terme *i* une fois et un document D_2 le cite quatre fois, alors D_2 doit être considéré plus pertinent que D_1, mais cela ne signifie pas que D_2 est quatre fois plus pertinent que D_1. Pour cette raison tf est souvent remplacée par $\log(1+tf)$ ou $\sqrt{tf}$ pour réduire l'impact des mots très fréquents et ainsi mieux refléter leurs poids au sein du document.

La fréquence df_i quantifie l'information apportée par un terme *i*. Un terme fréquemment répété dans un même document est généralement un bon candidat pour représenter le contenu de ce dernier. Cependant un terme qui apparaît fréquemment dans tous les documents de la collection ne représente pas un critère efficace de recherche dans cette dernière. Par exemple, le terme « ordinateur » serait largement utilisé dans une collection de documents informatiques et ce mot ne permet donc pas de distinguer le contenu de chaque document.

Le nombre de documents où le terme *i* apparaît, df_i, donne aussi une idée sur la valeur discriminatoire ou la précision sémantique de ce dernier. Un terme sémantiquement précis, par exemple « *ADN* », peut apparaître plusieurs fois dans un même document ou pas du tout dans un autre, tandis qu'un terme général, par exemple le verbe « *faire* », apparaît pratiquement avec la même fréquence dans tous les documents. La fréquence df est très utilisée dans les fonctions de pondération dont la plus connue est « la fréquence dans les documents inversée » (*inverse document frequency* désormais *IDF*). L'inverse de df améliore le poids des termes sémantiquement précis, par exemple, un terme qui apparaît 5 fois dans un seul document aura un poids plus important qu'un autre terme qui apparaît une seule fois dans 5 documents différents.

Des exemples de fonctions de pondération *w* utilisant le IDF :

$$w = \frac{tf}{df}$$

$$w = \frac{cf}{df}$$

$$w = \log(\frac{N}{df}) + 1$$

$$w = \begin{cases} (1 + \log(tf)) \log(\frac{N}{df}) & \text{si } tf \geq 1 \\ 0 & \text{si } tf = 0 \end{cases}$$

où N est le nombre de documents de la collection.

Généralement, les fonctions de pondération combinent tf et df. Cependant les longs documents ont plus de chance d'utiliser les mêmes termes que la requête de recherche. On voudrait qu'un terme recherché ait plus de poids s'il est cité une fois dans un document de 100 mots que s'il est cité une fois dans un document de 200 mots. Pour cette fin, on ajoute à la fonction de pondération un facteur de normalisation, noté *norme* (formule 2.1). Ce facteur est la norme du vecteur-document composé des fréquences de ses termes.

(Formule 2.1) $$norme(doc) = \sqrt{\sum_{i=1}^{n} tf_{i,doc}^2}$$

Selon Dumais [6], la fonction de pondération peut être vue comme le produit de deux facteurs, le facteur d'une pondération locale, noté *L*, et le facteur d'une pondération globale, noté *G*. Le premier sert à évaluer le poids du terme au sein du document où il apparaît, tandis que le deuxième mesure l'importance du terme dans toute la collection. Nous avons adopté cette représentation pour définir notre fonction de pondération notée « *Poids* », pour un document *doc* :

$$L = \log(tf + 1) \qquad \text{et} \qquad G = \log(N / Df) + 1$$

(Formule 2.2)

$$Poids = \frac{L * G}{norme(doc)}$$

2.2.6 Calcul de la similarité

La similarité a pour but d'évaluer la présence ou l'absence de points communs entre les documents. Les termes en commun ainsi que leurs distributions sont des indices révélateurs du niveau de similarité de leurs documents. Luhn [16] définit le concept de similarité:

« The more two representations agreed in given elements and their distribution, the higher would be the probability of their representing similar information. »

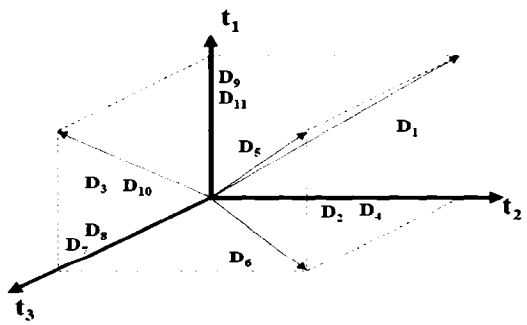

Figure 2.3 : Exemple de vecteur-document dans un espace à trois dimensions

Dans le *MEV*, les documents et les requêtes sont représentés par des vecteurs dans un espace multidimensionnel où chaque terme de l'index est une dimension de l'espace, figure 2.3.

Il existe plusieurs mesures pour estimer la similarité entre un document $Doc(d_1, d_2, d_3, ..., d_n)$ et une requête $Rq(q_1, q_2, q_3, ..., q_n)$ où d_i et q_i seront les poids correspondant au terme *i*. Les fonctions les plus fréquents sont le produit scalaire, la distance euclidienne, la corrélation entre les coordonnées des vecteurs ou encore le *cosinus* de l'angle formé par les deux vecteurs, nous rappelons ici leurs formules :

Le produit scalaire de deux vecteurs :

$$Sim(Doc, Rq) = Doc \bullet Rq$$

$$Sim(Doc, Rq) = \sum_{i=1}^{n} d_i * q_i$$

La distance euclidienne dans un espace à *n* dimensions :

$$Dist(Doc, Rq) = \sqrt{\sum_{i=1}^{n} (d_i - q_i)^2}$$

La corrélation avec la formule de Spearman où *N* est le nombre de dimensions et *r(x)* est le rang de *x* dans la distribution :

$$Spearman(x, y) = 1 - 6\frac{\sum_{i=1}^{N} (r(x_i) - r(y_i))^2}{N(N^2 - 1)}$$

Le *cosinus* de l'angle entre deux vecteurs :

(Formule 2.3) $$\cos(\overrightarrow{Doc}, \overrightarrow{Rq}) = \frac{Doc \bullet Rq}{\|Doc\| * \|Rq\|}$$

Rehder [24] a comparé l'efficacité de ces mesures dans le cadre d'une expérience d'acquisition de nouvelles connaissances [12]. Il a démontré que ces mesures sont liées et qu'il y a un grand niveau de corrélation entre eux. Il a trouvé que le *cosinus* réussit à refléter les liens sémantiques mieux que le reste des mesures étudiées.

Il existe d'autres formules normalisées de mesure de similarité comme la formule de « *Dice* » et la formule de « *Jaccard* » [25]. Cependant la formule du *cosinus* est la plus populaire dans la littérature de la *RI*.

$$Dice = \frac{2 * \sum_{i=1}^{n} tf_d * tf_q}{\sum_{i=1}^{n} tf_d^2 + \sum_{i=1}^{n} tf_q^2} \qquad Jaccard = \frac{\sum_{i=1}^{n} tf_{di} * tf_{qi}}{\sum_{i=1}^{n} tf_{di}^2 + \sum_{i=1}^{n} tf_{qi}^2 - \sum_{i=1}^{n} tf_{di} * tf_{qi}}$$

Le *MEV* est basé sur le nombre d'occurrence et la distribution des termes dans le document et dans la collection. Ainsi, il se limite à un calcul des statistiques et n'évalue pas la valeur sémantique des mots. On voudrait doter le système de RI d'une capacité d'interprétation du sens des mots. Dans le chapitre suivant, nous présentons le modèle d'analyse sémantique latente (*LSA*) qui, en plus de reprendre les bases du *MEV*, offre une nouvelle méthode pour approcher la sémantique des termes. Cette nouvelle approche a donné un nouveau souffle aux applications basées sur la sémantique des mots tel que la catégorisation des documents ou encore la cotation des études de cas.

Chapitre 3 : Analyse sémantique latente

3.1 Introduction

L'analyse sémantique latente se base sur le modèle d'espace vectoriel et reprend le principe d'analogie entre la proximité spatiale des vecteurs et la proximité sémantique des documents. *LSA* apporte une solution à deux problèmes connus dans la RI. Le premier est « la synonymie », plusieurs mots du même vocabulaire expriment le même concept comme « voiture », « automobile » ou « véhicule ». Le deuxième est « la polysémie », un mot isolé n'a pas de sens précis, il faut le placer dans un contexte. Ainsi le mot « avocat » dans document *A* peut faire référence à la profession d'avocat et dans un autre document *B* au fruit de l'avocatier. Se baser le simple terme « avocat » pour évaluer la similarité de ces deux documents ne reflétera pas le vrai thème des documents.

Les humains peuvent comprendre le sens d'un nouveau mot en se basant sur son contexte. C'est d'ailleurs de cette manière que les jeunes apprennent la majorité des nouveaux mots et non grâce aux dictionnaires. L'analyse sémantique latente (*LSA*) s'inscrit dans ce cadre en cherchant à déduire le sens des termes selon leurs contextes.

> *"Latent Semantic Analysis (LSA) is a theory and method for extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of text"* [12]

LSA se base sur le principe que plus les mots apparaissent ensemble ou dans les mêmes contextes, plus probable est le lien sémantique entre ces mots. C'est ce qu'on appelle les associations directes et d'ordre supérieur. Ainsi on peut déduire le sens d'un mot selon son contexte et, plus généralement, le sens d'un document selon les mots qu'il contient.

L'association dite « *directe* », ou « *de premier ordre* », fait référence à la présence simultanée de deux termes dans le même document ou contexte. L'association « *d'ordre supérieur* » pointe la liaison entre des termes qui n'apparaissent pas ensemble mais sont reliés à travers une combinaison d'associations directes. On distingue plusieurs niveaux d'associations d'ordre supérieur. Sur la figure 3.1, les termes *A* et *C* n'apparaissent pas ensemble, cependant ils sont liés par une association de niveau 2 par le biais de *B*. La même chose pour les termes A et D qui sont liés par une association de niveau 3 à travers *B* et *C*, et ainsi de suite.

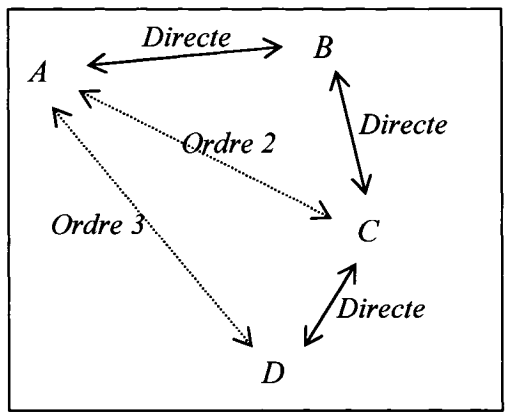

Figure 3.1 : Niveaux des associations entre termes

Cette nouvelle approche a fait le succès de *LSA* dans plusieurs domaines. Elle est évaluée pour classifier les documents, simuler le raisonnement humain pour les réponses à choix multiples, évaluer le contenu et la cohérence des documents, recommander des lectures selon les connaissances du lecteur et aussi pour coter les essais des étudiants qui est le sujet de ce mémoire [13][14].

3.2 Fonctionnement de LSA

LSA utilise les mêmes étapes que le *MEV* en plus d'appliquer une réduction de l'espace vectoriel. Cette réduction permet d'extraire les associations d'ordre supérieur en plus de concentrer l'information de la matrice des fréquences dans un espace ayant un nombre réduit de dimensions. *LSA* utilise une technique de décomposition matricielle appelée la division en valeurs singulières (*SVD*). Cette technique permet de réduire les dimensions de la matrice en la projetant sur les dimensions les plus importantes.

Pour commencer, la technique *LSA* doit être entraînée sur un grand corpus de texte relatif au domaine en question. La technique déduit le sens des termes à travers les exemples et les contextes de leurs utilisations. Le but de cette étape est de doter *LSA* d'une base de connaissance solide lui permettant de bien assimiler les notions et le vocabulaire du domaine en question. Généralement on utilise des encyclopédies ou des livres de référence car les concepts y sont bien expliqués et répétés dans différents contextes. Par exemple, dans le carde des expériences de Landauer et Dumais [12], ils ont utilisé une collection de 4,6 million de mots tirés de l'encyclopédie « Grolier's Academic American Encyclopedia ». Plus le corpus est riche, plus les liaisons entre les termes sont claires et fortes grâce aux associations de niveaux supérieurs.

Par la suite, on représente l'ensemble de ces entrées dans une grande matrice M $(n \times p)$, les n lignes correspondent aux termes et les p colonnes correspondent aux documents ou articles, chaque cellule $M[i, j]$ contient le nombre de fois que le terme de la ligne i apparaît dans le document de la colonne j. On rappelle ici que toutes les variantes d'un mot, appelées *terme,* sont représentées par une seule ligne. La taille de cette matrice dépend de la taille et la quantité de document disponible, par exemple dans l'expérience de Landauer [12], la matrice se compose de 60,768 lignes et 30,473 colonnes.

Avant d'appliquer la *SVD*, il faut d'abord pondérer les termes. La fréquence dans les documents inversée « *IDF* » est généralement utilisée pour évaluer les fréquences des termes et ainsi mieux refléter leurs poids dans le document et dans la collection (formule 2.1). Grâce à la *SVD*, on peut réduire le nombre des dimensions d'une matrice $M(n\times p)$ à un nombre k dans l'intervalle $[1, min(n, p)]$.

La *SVD* est une méthode mathématique souvent utilisée pour calculer le rang d'une matrice, trouver le repère d'un espace ou encore pour calculer l'inverse de matrices surtout les matrices singulières (c'est-à-dire celles dont le déterminant est nul).

Rappelons ici quelques propriétés importantes des matrices, soient A et B deux matrices et A^{-1} et B^{-1} leurs inverses respectifs :

Formule 3.1 $AA^{-1} = I$ (matrice identité)

Formule 3.2 $(AB)^{-1} = B^{-1}A^{-1}$

Formule 3.3 $t(AB) = t(B)t(A)$

Formule 3.4 $t(A^{-1}) = t(A)^{-1}$

Formule 3.5 Si A est diagonale $[diag(A_{i,i})]^{-1} = diag(1/A_{i,i})$

Formule 3.6 A est orthogonale si $A^{-1} = t(A)$

Formule 3.7 Si $det(A) = 0$ alors A^{-1} n'existe pas, A est *singulière*

Formule 3.8 Si $A(x\times y)$ alors $rang(A)\leq min(x,y)$

La *SVD* permet de décomposer la matrice des fréquences $M(n\times p)$ en trois facteurs uniques, deux matrices $U(n\times r)$ et $V(p\times r)$ orthogonales et un vecteur D tel que :

Formule 3.9 $M(n\times p) = U(n\times r) * diag(D)(r\times r) * t(V(p\times r))$

où le nombre *r* est le rang de la matrice *M,* et *t()* la fonction transposée matricielle, et *diag (D)* est une matrice diagonale ayant pour valeurs le vecteur *D*.

Les éléments du vecteur *D* sont les valeurs singulières de *M* triées en ordre décroissant tel que $d_1 \geq d_2 \geq ... \geq d_r \geq d_{r+1} = ... = d_n = 0$. Cet ordre correspond à l'ordre d'importance des dimensions.

SVD est testée dans plusieurs domaines de recherche. Par exemple pour éliminer les bruits de signaux ou les imprécisions des mesures. À l'instar des autres domaines, la recherche d'information l'utilise pour supprimer les interférences ou les imprécisions sémantiques.

Dans la formule 3.9, si on remplace *r* par un nombre *k* tel que $k \leq r$ alors la matrice construite, notée M_k, est la matrice de rang k la plus proche de *M* dans le sens des moindres carrées (c'est-à-dire que la valeur de $\sum_{i=1}^{m}\sum_{j=i}^{n}(M[i,j]-M_K[i,j])^2$ est à son minimum) (formule 3.10) Cette nouvelle matrice M_k représente les *k* dimensions les plus importantes et élimine les faibles dimensions. Elle est donc sémantiquement plus précise que M.

Formule 3.10 $$M_k(n\times p) = U_k(n\times k)\ diag(D_k)(k\times k)\ t(V(p\times k))$$

On peut comparer deux documents dans la matrice initiale M avec le produit scalaire de leurs vecteurs. Cette méthode est valide pour la matrice M_k aussi :

$$t(M_k)\bullet M_k = t(U_k diag(D_k) t(V_k))\bullet U_k diag(D_k) t(V_k)$$

En appliquant la formule 3.3 :

$$t(M_k)\bullet M_k = V_k t(diag(D_k)) t(U_k)\bullet U_k diag(D_k) t(V_k)$$

Puisque D est diagonale alors $diag(D) = D$

On simplifie la formule résultat en utilisant les formules 3.4 et 3.6.

Formule 3.11 [8] $t(M_k) \bullet M_k = V_k diag(D_k) \bullet t(V_k diag(D_k))$

On représente un nouveau document ou une requête Q par la somme de ses termes dans l'espace M, la nouvelle colonne est notée $M[q]$. Cette dernière correspond à une nouvelle ligne dans la matrice V qu'on note V[q], tel que :

$$M[q] = U\ diag(D)\ t(V[q])$$

On isole le vecteur $t(V[q]) = diag(D)^{-1}\ U^{-1}\ M[q]$

Formule 3.12 [8] $V[q] = t(M[q])\ U\ diag(D)^{-1}$

L'espace réduit, appelé aussi espace sémantique, offre une meilleure représentation du contenu des documents car il permet de ressortir les liaisons latentes entre ces derniers. Les comparaisons basées sur l'espace réduit sont plus efficaces et consomment moins de ressource que dans l'espace initial.

Notons ici que si on garde un grand nombre de dimensions, l'information ne sera pas bien concentrée et la réduction n'aura pas un grand impact sur les résultats. Par contre si on garde très peu de dimensions, on risque de perdre trop d'informations utiles et ainsi détériorer les résultats [19]. Il n'y a pas de façon directe pour préciser la valeur exacte de *k*. Pour cela on passe par des tests d'ajustement et on garde les valeurs qui produisent les meilleurs résultats. Ces valeurs sont généralement entre 50 et 400 dimensions [1], par exemple, pour un corpus dont la matrice originale est composée de 60.768 lignes et 30 473 colonnes, les meilleures valeurs de *k* sont entre 300 et 325 [12] .

3.3 Exemple

Pour illustrer le fonctionnement de *LSA*, on reprend l'exemple de Landauer et ses co-équipiers dans *« Introduction to LSA »* [13]. On considère une collection de neuf titres d'articles techniques, cinq d'entre eux relèvent du domaine d'interaction humain-ordinateur (noté groupe *c*) et les quatre autres de la théorie des graphes (noté groupe *m*). Dans cet exemple, on garde seulement les termes qui apparaissent au moins deux fois dans les titres (les termes en gras), le tableau 3.1 recense les mots avec leurs fréquences dans chaque titre.

c1 : ***Human*** machine ***interface*** for ABC ***computer*** applications
c2 : A ***survey*** of ***user*** opinion of ***computer system response time***
c3 : The ***EPS user interface*** management ***system***
c4 : ***System*** and ***human system*** engineering testing of ***EPS***
c5 : Relation of ***user*** perceived ***response time*** to error measurement
m1: The generation of random, binary, ordered ***trees***
m2: The intersection ***graph*** of paths in ***trees***
m3: ***Graph minors*** IV: Widths of ***trees*** and well-quasi-ordering
m4: ***Graph minors***: A ***survey***

Figure 3.2 : Titres de l'exemple de* LSA *[13]

Tableau 3.1 : Matrice des fréquences des termes [13]

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

Calculons la corrélation entre deux vecteurs en utilisant la formule de *Spearman* :

$$Spearman(x,y) = 1 - 6\frac{\sum_{i=1}^{N}(r(x_i) - r(y_i))^2}{N(N^2-1)}$$

Notons ici les corrélations entre le vecteur du mot « *humain* » et les vecteurs des mots « *user* » et « *minors* » :

Spearman (*humain, user*) = -0,38 et Spearman (*humain,minors*) = -0,29

La réduction de l'espace vers k dimensions consiste à mettre à zéro toutes les valeurs du vecteur D sauf les k premières (noté D_k), et à ne considérer que les k premières colonnes de la matrice U (noté U_k) et les k premières lignes de la matrice $t(V)$ (noté V_k). La matrice M' construite par la multiplication de ces trois nouveaux facteurs est la matrice de rang k la plus proche de M en termes des moindres carrés, figure 3.3.

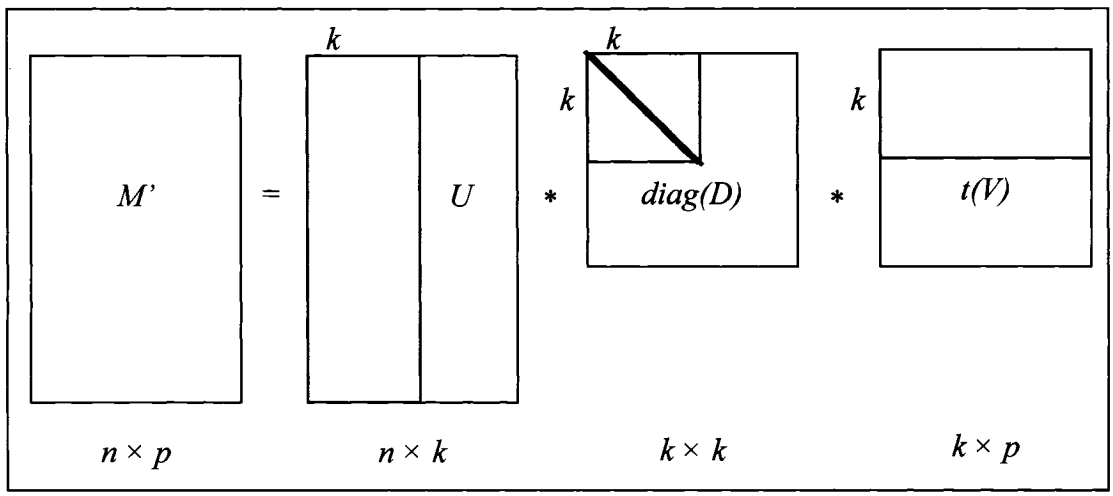

***Figure 3.3 : Réduction des dimensions de l'espace avec la* SVD.**

U_k, appelée la matrice des mots, a autant de lignes que la matrice M et représente les mots dans le nouvel espace à k dimensions. De même, les lignes de V_k, la matrice des documents, représentent les documents dans l'espace de réduit.

Une bonne projection, d'un espace multidimensionnel sur un espace réduit, réussit à garder le plus d'information possible, ceci est équivalent à garder les dimensions qui captent le plus de variations et d'éliminer le reste.

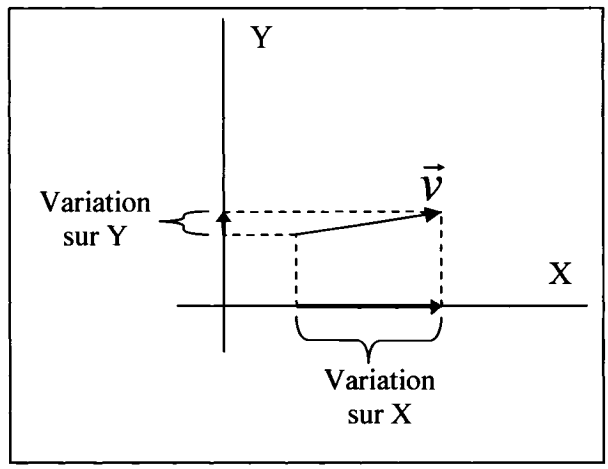

Figure 3.4 : Exemple de projection

Supposons qu'on a un vecteur $\vec{v}$ représenté dans un espace à deux dimensions (X, Y) (figure 3.4). On veut réduire cet espace à une seule dimension avec la moindre perte d'information possible (la taille et le sens du vecteur). D'après le schéma, l'axe (Y) capte très peu d'information (petite taille et une grande déviation du sens du vecteur) alors que la projection sur l'axe (X) est proche du vecteur initial. Donc la représentation avec la moindre perte nous amène à éliminer l'axe (Y) et à garder l'axe (X).

La *SVD* retourne une liste de mesures de l'importance des dimensions triée en ordre décroissant selon la quantité de variation captée sur chaque dimension. Cette liste est la diagonale de la matrice *D*. Durant la projection, garder les *k* premières valeurs de la matrice *D* est équivalent à projeter sur les *k* meilleures dimensions de l'espace vectoriel.

Dans le cadre de l'exemple, les résultats de la décomposition en valeurs singulières sont :

U (12×9) =

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

D (12×9) =

3.34								
	2.54							
		2.35						
			1.64					
				1.50				
					1.31			
						0.85		
							0.56	
								0.36

t (V (12×9)) =

0.20	0.61	0.46	0.54	0.28	0.00	0.01	0.02	0.08
-0.06	0.17	-0.13	-0.23	0.11	0.19	0.44	0.62	0.53
0.11	-0.50	0.21	0.57	-0.51	0.10	0.19	0.25	0.08
-0.95	-0.03	0.04	0.27	0.15	0.02	0.02	0.01	-0.03
0.05	-0.21	0.38	-0.21	0.33	0.39	0.35	0.15	-0.60
-0.08	-0.26	0.72	-0.37	0.03	-0.30	-0.21	0.00	0.36
0.18	-0.43	-0.24	0.26	0.67	-0.34	-0.15	0.25	0.04
-0.01	0.05	0.01	-0.02	-0.06	0.45	-0.76	0.45	-0.07
-0.06	0.24	0.02	-0.08	-0.26	-0.62	0.02	0.52	-0.45

La projection sur deux dimensions consiste à garder juste les deux premières colonnes de chaque matrice (la partie grisée). On obtient le même résultat en affectant 0 à toute la diagonale de la matrice *D* excepté les *k* premières. La nouvelle matrice de l'espace réduit est :

$$M'(12\times10) = U(12\times2) * diag(D)(2\times2) * t(V(12\times2))$$

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	0.16	0.40	0.38	0.47	0.18	-0.05	-0.12	-0.16	-0.09
interface	0.14	0.37	0.33	0.40	0.16	-0.03	-0.07	-0.10	-0.04
computer	0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
user	0.26	0.84	0.61	0.70	0.39	0.03	0.08	0.12	0.19
system	0.45	1.23	1.05	1.27	0.56	-0.07	-0.15	-0.21	-0.05
response	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
time	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
EPS	0.22	0.55	0.51	0.63	0.24	-0.07	-0.14	-0.20	-0.11
survey	0.10	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.42
trees	-0.06	0.23	-0.14	-0.27	0.14	0.24	0.55	0.77	0.66
graph	-0.06	0.34	-0.15	-0.30	0.20	0.31	0.69	0.98	0.85
minors	-0.04	0.25	-0.10	-0.21	0.15	0.22	0.50	0.71	0.62

On remarque que la projection a changé la valeur de chaque cellule de la matrice. Le mot « *tree* » n'apparaît pas dans le titre *m4* et avait donc un poids nul dans la matrice initiale, alors que dans l'espace réduit le zéro est changé pour 0,66. Ceci signifie que même si le terme est absent, son contexte amène à le considérer présent dans l'espace réduit (« *tree* » apparaît dans le même contexte que « *graph* » et « *minors* »)

Pour le même titre *m4*, la valeur de « *survey* » était à 1 puis devenu 0,44 pour dire que ce terme a peu de chance d'être présent dans des contextes similaires à *m4*.

On recalcule la corrélation entre «*human-user*» et «*human-minors*» dans l'espace réduit :

Spearman (*human, user*) = 0,94 et Spearman (*human, minors*) = -0,83

Malgré que ces trois termes n'apparaissent pas dans les mêmes titres, *LSA* a trouvée que « *human* » et « *user* » sont très similaires (dans le domaine des interactions homme-machine ces deux termes sont pratiquement équivalents) et que « *human* » et « *minors* » n'ont pas de liens communs. *LSA* s'appuie sur ce qu'on appelle les liaisons latentes ou relations d'ordre supérieur (voir l'introduction de ce chapitre).

On calcule la corrélation entre les titres (les colonnes) avant et après la réduction de l'espace, les tableaux 3.2 et 3.4 résument les résultats :

Tableau 3.2 : Corrélation entre les titres avant la réduction des dimensions

	c1	c2	c3	c4	c5	m1	m2	m3
c2	-0.19							
c3	0.00	0.00						
c4	0.00	0.00	0.47					
c5	-0.33	0.58	0.00	-0.31				
m1	-0.17	-0.30	-0.21	-0.16	-0.17			
m2	-0.26	-0.45	-0.32	-0.24	-0.26	0.67		
m3	-0.33	-0.58	-0.41	-0.31	-0.33	0.52	0.77	
m4	-0.33	-0.19	-0.41	-0.31	-0.33	-0.17	0.26	0.56

Tableau 3.3 : Moyennes des corrélations par groupe avant la réduction des dimensions

	Groupe c	Groupe m
Groupe c	0,02	-0,30
Groupe m	-0,30	0,44

Les valeurs 0,02,-0,30 et 0,44 sont les moyennes de corrélation entre les groupes de titres, respectivement, entre les titres du groupe *c*, entre les titres du groupe *c* et ceux du groupe *m* et enfin entre les titres du groupe *m*. On remarque que les corrélations ne sont pas assez fortes pour ressortir clairement les liens entre les groupes.

Tableau 3.4 : Corrélation entre les titres dans l'espace réduit

	c1	c2	c3	c4	c5	m1	m2	m3
c2	0.91							
c3	1.00	0.91						
c4	1.00	0.88	1.00					
c5	0.85	0.99	0.85	0.81				
m1	-0.85	-0.56	-0.85	-0.88	-0.45			
m2	-0.85	-0.56	-0.85	-0.88	-0.44	1.00		
m3	-0.85	-0.56	-0.85	-0.88	-0.44	1.00	1.00	
m4	-0.81	-0.50	-0.81	-0.84	-0.37	1.00	1.00	1.00

On refait le même calcul de corrélation avec la matrice de l'espace réduit, on remarque que les nouvelles valeurs reflètent la relation entre les groupes de titre, et que les moyennes de corrélation dans l'espace réduit sont très révélatrices :

Tableau 3.5 : Moyennes des corrélations par groupe dans l'espace réduit

	Groupe c	Groupe m
Groupe c	0,92	
Groupe m	-0,72	1,00

On voit clairement que les titres du même groupe ont une corrélation presque parfaite ($\approx$ 1) alors que la valeur -0,72 confirme qu'il n'y a pas de similarité entre le thème d'interaction humain ordinateur et la théorie des graphes.

Cet exemple est simpliste : notamment la taille du corpus est trop petite et on n'a pas fait de pondération ni de test pour trouver le meilleur nombre de dimensions pour la réduction d'espace. Dans ce qui suit nous présenterons deux expériences qui ont utilisé *LSA* à une échelle réelle.

3.3 LSA et l'acquisition des nouvelles connaissances

Les expériences de Landauer et Dumais [12] répondent à un problème largement connu dans le domaine de la philosophie et des sciences : comment expliquer qu'une personne a plus de connaissance que ce que lui a été enseigné ? Autrement dit comment arrive-t-on à comprendre des nouveaux mots qu'on a jamais vu dans le passé ?

La réponse apportée par Landauer et Dumais [12] est que le contexte d'utilisation explique les nouveaux termes. Pour simuler le processus de jugement de similarité entre les mots, *LSA* a été utilisée pour répondre à une partie du test de langue (test par synonymes) identique à celui de l'admission des étudiants d'origine non anglophones aux universités américaines.

LSA est d'abord entraînée sur un large corpus de 4,6 million de mots tiré de l'encyclopédie « Grolier's Academic American Encyclopaedia ». Un ensemble de 30 473 articles est utilisé pour construire une matrice de 30 473 colonnes et 60 768 lignes, chaque colonne représente un article, chaque ligne représente le format canonique unique de chaque famille de terme. Chaque cellule de la matrice représente le nombre de fois que le mot de sa ligne apparaît dans le document de sa colonne. D'ailleurs seuls les mots ayant une fréquence globale supérieure à deux sont retenus dans cette matrice.

Par la suite, la pondération des termes est appliquée avec la formule suivante :

$$w = \frac{\mathrm{Ln}(1+df)}{cf}$$

Après une série de tests, le nombre de dimensions de l'espace réduit est fixé à 300, la figure 3.5 explique ce choix et démontre l'effet du nombre de dimensions sur l'exactitude des réponses obtenues. On y remarque que garder un grand nombre de dimensions cause une dispersion d'information et que garder peu de dimensions implique une grande perte d'information, dans les deux cas la projection n'est pas à son optimum.

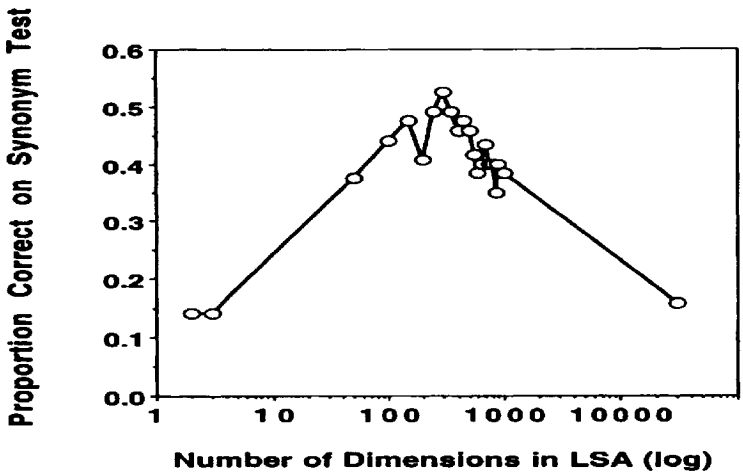

Figure 3.5 : L'impact du nombre de dimensions sur la qualité des réponses [12]

Pour le test de langue, *LSA* utilise l'espace réduit pour trouver le meilleur équivalent sémantique d'un terme parmi quatre choix. *LSA* calcule la similarité avec le *cosinus* entre le terme en question et chacun des choix et retient le choix le plus proche.

Au terme de ce test, *LSA* obtient un score de 51.5 points (64.4% de bonnes réponses) ce qui est équivalent à la moyenne de passage pour des étudiants non anglophones postulant pour les examens d'admission aux universités américaines.

Par la suite, Landauer et Dumais [12] ont refait le même test en utilisant la matrice des fréquences (après la pondération et la décomposition) sans aucune réduction de dimension, le pourcentage des bonnes réponses était de 15.8%, ce qui signifie que *LSA* est nettement plus performante avec la réduction de l'espace sémantique.

3.4 LSA et la recommandation des lectures

Pour que l'apprentissage par lecture soit efficace, les textes de lecture doivent utiliser des connaissances déjà acquises par l'étudiant pour introduire des nouvelles connaissances. *LSA* est utilisée pour estimer la quantité d'information qu'un étudiant apprendrait d'un texte, et aussi pour recommander des lectures selon le niveau des connaissances incluses dans le document et le niveau de l'étudiant. Wolfe et ses co-équipiers [33] ont exploré cette idée avec le thème de la cardiologie et la circulation sanguine.

Dans cette expérience, Les participants doivent d'abord répondre à un test pour évaluer leurs connaissances dans le domaine de la cardiologie. En suite, ils lisent un article référence parmi quatre (*A, B, C* ou *D*). Les articles sont triés en ordre croissant du niveau de leurs contenus du niveau élémentaire au niveau universitaire. Enfin, les participants répondent au même test encore une fois.

Les résultats ont démontré que les participants apprennent plus lorsque le texte n'est ni trop difficile (des concepts tout à fait nouveaux) ni trop facile (des concepts déjà connus).

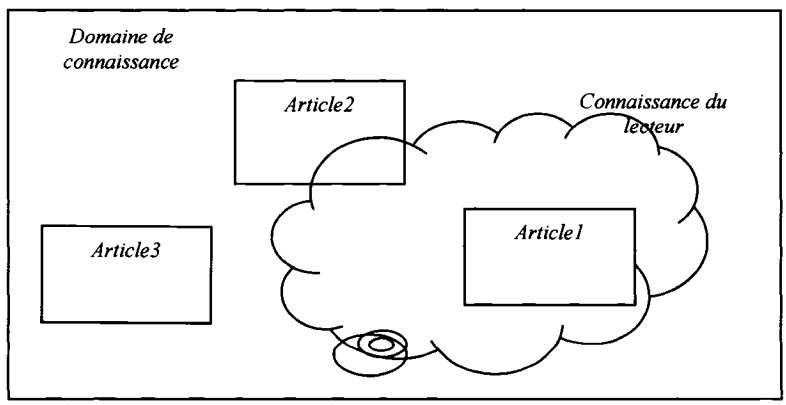

Figure 3.6 : Niveaux de connaissance du lecteur et des articles de lecture

Dans la figure 3.6, on voit que l'article 2 est la meilleure lecture pour le lecteur car ce dernier a les connaissances de base pour le comprendre.

Dans cette expérience, un espace sémantique de 100 dimensions est construit avec 36 articles citant 17 880 mots dont 3 034 mots uniques. Par la suite, on représente chaque article référence par un vecteur « *article-référence* », on représente l'essai d'étudiant avant et après la lecture par les vecteurs « *test-avant-lecture* » et « *test-après-lecture* ».

Le *cosinus* est utilisé comme mesure de similarité entre les vecteurs. La comparaison des lectures considère que si le cosinus entre le « *essai-avant-lecture* » et « *article-référence* » est trop petit ou nul (article 3 de la figure 3.3), alors le participant n'a pas assez de connaissances pour comprendre l'article en question. Par contre, si la similarité est très grande (article 1 sur la figure 3.3), alors le participant connaît déjà les notions traitées dans la lecture. Dans ces deux cas, la lecture ne permet pas un niveau d'apprentissage optimal des nouvelles connaissances. La lecture la plus profitable pour un participant serait celle ayant une valeur de *cosinus* moyenne avec son « *essai-avant-lecture* » (article 2 sur la figure 3.3).

La figure 3.4 montre la valeur moyenne du cosinus entre l'article lu d'un coté et les essais avant et après lecture de l'autre coté. On peut ainsi évaluer les connaissances acquises à travers la lecture. On remarque que la similarité entre les essais avant-lecture et les articles de référence diminue au fur et à mesure que le niveau de complexité des articles monte. Deuxième remarque, les participants qui ont lu un des articles *A, B* ou *C* ont une bonne similarité après-lecture c'est-à-dire qu'ils ont réussi à comprendre et reproduire le contenu de l'article lu. La seule exception est l'article D. Ceci s'explique du fait que le contenu de cet article est difficile à comprendre et que les participants n'ont pas assez de bagage pour comprendre ses concepts.

On peut remarquer que les participants ayant lu l'article C ont acquis plus de connaissance que les autres. L'écart entre les deux degrés de similarité avant et après la lecture confirme l'hypothèse de départ qui suppose qu'une lecture bien adaptée peut améliorer le processus d'apprentissage.

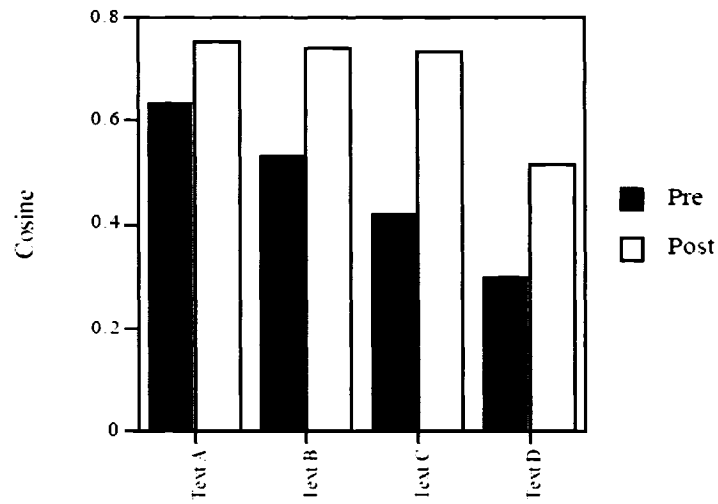

Figure 3.7 : La similarité lecture-test avant et après la lecture [33].

Cette étude a démontré que *LSA* réussit à quantifier correctement le niveau de connaissance des documents et ainsi elle peut recommander des lectures dépendamment du niveau de l'étudiant et ainsi optimiser le niveau d'apprentissage.

Dans ces deux expériences (acquisition des nouvelles connaissances et recommandation des lectures), *LSA* a fait preuve d'une grande efficacité et elle a démontré un bon niveau de sensibilité au sens des mots. Dans le prochain chapitre, il est question d'évaluer *LSA* pour la correction d'analyse de cas. Nous voulons mesurer à quel point cette technique est sensible à la qualité et la sémantique des essais.

Chapitre 4 : La correction d'analyse de cas

4.1 Introduction

Au cours des chapitres précédents, nous avons présenté le domaine de la recherche d'information (RI) ainsi que ses modèles les plus connus. On s'est particulièrement intéressé au modèle d'espace vectoriel (*MEV*) qui est à la base de l'analyse sémantique latente (*LSA*). La présentation du *MEV* nous a permis d'expliquer les étapes importantes de *LSA* tel que la préparation de textes, la pondération des *termes* et le concept de similarité entre documents. Par la suite, nous avons détaillé le fonctionnement de *LSA* et nous avons présenté deux exemples de ses applications. Dans ce chapitre nous abordons le sujet principal de ce mémoire qui est la cotation automatique des essais.

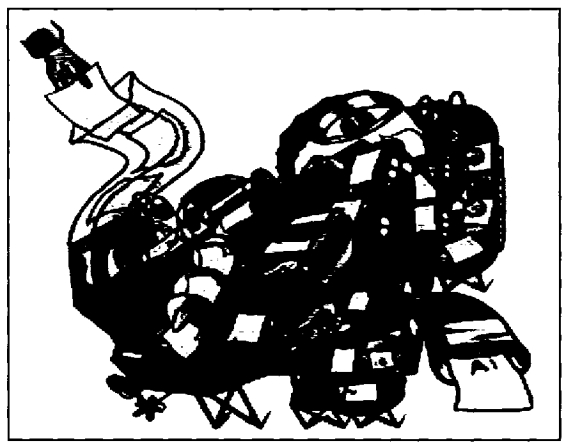

Figure 4.1 : Schéma caricatural de la correction automatique (*Robert Soulé*)

L'idée de cotation des essais par ordinateur était explorée pour la première fois durant les années soixante. Depuis cette idée a attiré beaucoup d'attention, surtout ces dernières années grâce à l'utilisation à grande échelle des ordinateurs dans la rédaction et la remise des travaux par les étudiants et les professeurs.

La correction automatique des essais permettrait de libérer les professeurs de la lourde tâche répétitive de corriger manuellement les essais mais, avant tout, le but d'une telle application est d'assister les étudiants pour améliorer la qualité de leurs essais et ainsi profiter d'une correction rapide, neutre et disponible en tout temps.

Les premières expériences [21] dans ce domaine datent des années 60. À ce stade on cherchait à juger la qualité des essais par de simples mesures comme la longueur des phrases, le nombre des mots, le nombre des propositions, etc. Ces mesures se basent sur des propriétés lexicales et syntaxiques superficielles et ne jugent pas le contenu proprement dit. *LSA*, avec son espace sémantique, offre de nouvelles options. En se basant sur des techniques statistiques plus poussées, *LSA* extrait une dimension du texte qu'on qualifie de « sémantique ». Désormais, on ne veut plus juger les mots mais les idées derrières ces mots.

Dans ce chapitre, nous commençons par introduire les premières tentatives dans ce domaine, expliquer les idées de base et donner des références et des résultats des tests.

4.2 Systèmes de correction automatique

4.2.1 Project essay grade

L'article de Page [21], publié en 1966, a déclenché le débat sur la possibilité qu'un ordinateur corrige correctement les essais textes des étudiants. Après deux années de recherche, Page a estimé que son système de correction automatique appelé PEG (*Project Essay Grade)* peut corriger les essais aussi bien qu'un correcteur humain.

Depuis le début, Page a fait la différence entre une évaluation sémantique du contenu et une évaluation du style et du lexique. À défaut de pouvoir juger l'aspect sémantique, Page s'est intéressé à l'évaluation de style en se basant sur des critères facilement observables appelés « *proxies* », Parmi ces derniers on cite la longueur moyenne des phrases, nombre de paragraphes, les mots courants (une variante du '*stop list*'), les fautes d'orthographe, la ponctuation et les caractères spéciaux (points, virgules, apostrophes, parenthèses, traits d'union). Au total, Page utilise une trentaine de critères.

La méthode de Page a besoin d'un corpus d'entraînement. Ce dernier doit se composer de copies corrigées répondant au même énoncé que les essais à corriger. Le but est de trouver parmi les « *proxies* » ceux dont les mesures permettent de prédire les notes attribuées par les correcteurs humains. Les « *proxies* » sélectionnés sont combinés, avec des coefficients ajustés, pour approcher le mieux possible les notes des correcteurs humains.

La formule obtenue est utilisée pour noter le reste des copies appelé « ensemble de validation ». Par la suite, la qualité de la correction automatique est estimée selon sa corrélation avec la correction humaine.

Dans une de ses premières et rares expériences divulguées, Page [32] a utilisé 276 copies d'étudiants d'une école secondaire. Chaque copie est corrigée de manière *holistique* par quatre correcteurs indépendants. La cotation holistique consiste à attribuer une note globale à l'ensemble de l'essai sans considérer sa structure paragraphes ou en questions. La somme des quatre notes constituent la note finale de la copie. La moitié de ces copies (138) sont utilisées pour entraîner PEG. De cette façon, il définit les « *proxies* » pertinents pour cette expérience et il calcule leurs coefficients, la liste des variables proxies est présentée dans le tableau 4.1.

Les résultats de ces expériences étaient encourageants. PEG a obtenu une corrélation égale à 0,5 avec la somme des notes des correcteurs humains, ce qui est comparable au niveau de corrélation entre les correcteurs humains [32].

Dans une autre expérience plus récente (1994), Page [32] a rapporté des corrélations, avec un correcteur à la fois, variant entre 0,54 et 0,74, ce qui représente une nette amélioration par rapport aux premières expériences. PEG continue d'évoluer et d'améliorer son niveau de corrélation. Cependant, PEG est critiqué pour deux raisons principales. Premièrement, il utilise des critères facilement manipulables ce qui entraîne un risque de fraude, en plus d'être non-sémantiques. La deuxième raison est la nécessité de corriger manuellement une partie des essais à corriger pour les utiliser comme ensemble d'entraînement.

PEG est commercialisé par la société « *Tru-Judge* » d'où le manque des détails concernant les composantes de ce système.

Proxy variables	Correlation with essay score (human-rated)	Beta weights
Title present	0.04	0.09
Average sentence length	0.04	-0.13
Number of paragraphs	0.06	-0.11
Subject-verb openings	-0.16	-0.01
Length of essay in words	0.32	0.32
Number of:		
Parentheses	0.04	-0.01
Apostrophes	-0.23	-0.06
Commas	0.34	0.09
Periods	-0.05	-0.05
Underlined words	0.01	0.00
Dashes	0.22	0.10
Colons	0.02	-0.03
Semicolons	0.08	0.06
Quotation marks	0.11	0.04
Exclamation marks	-0.05	0.09
Question marks	-0.14	0.01
Prepositions	0.25	0.10
Connectives	0.18	-0.02
Spelling errors	-0.21	-0.13
Relative pronouns	0.11	0.11
Subordinating conjunctions	-0.12	0.06
Common words on Dale[a]	-0.48	-0.07
Sentences' end punctuation present	-0.01	-0.08
Hyphens	0.18	0.07
Slashes	-0.07	-0.02
Average word length in letters	0.51	0.12
Standard deviation of word length	0.53	0.30
Standard deviation of sentence length	-0.07	0.03

Tableau 4.1 : Variables « proxies » de PEG [32]

4.2.2 E-rater

E-rater, une version améliorée de PEG, a comme principe de base d'éviter toute mesure directe de la longueur de l'essai. Ainsi *E-rater* se base sur plus de cinquante mesures dont les quatre principales sont : la syntaxe, la cohérence du discours, le contenu et le lexique [4].

- L'analyse syntaxique consiste à découper le texte en phrases et mettre en évidence la structure syntaxique des phrases. Cette mesure permet d'évaluer la variété syntaxique utilisée par l'étudiant.

- La cohérence : Grâce à l'utilisation des méthodes heuristiques, E-rater essaie de détecter l'organisation des idées en marquant les mots spéciaux comme « premièrement, deuxièmement », « au début », ou des termes d'argumentation comme «car » ou « par ce que ». Cette étape permet de découper le texte en une suite d'arguments ou d'idées qui vont être évalués par la suite.
- Le contenu : E-rater suppose que, contrairement aux mauvais essais, les bons essais utilisent un vocabulaire sémantiquement précis et pertinent au sujet de l'examen. Ainsi on peut les identifier grâce aux mots clés utilisés et à la distribution des mots dans les essais. Le E-rater adopte le *MEV* comme modèle de recherche d'information et le « *cosinus* » comme mesure de similarité entre les essais et le corrigé.
- L'analyse lexicale : E-rater évalue des critères tels que le nombre de mots uniques utilisés et la longueur moyenne des mots. Ces mesures reflètent le style et la richesse du vocabulaire de l'étudiant.

Le « E-rater » est la propriété de *Educational Testing Service* (ETS) et il est utilisé par cette même société pour offrir un service de correction à plusieurs instituts.

4.2.3 Intelligent essay assessor (IEA)

Le *Intelligent Essay Assessor* (IEA) est un système de correction automatique basé sur *LSA* [14]. Avant de corriger les essais des étudiants, il faut d'abord entraîner IEA avec un échantillon de textes relié au domaine en question et lui fournir des exemples de copies corrigées. IEA offre la possibilité de corriger les essais en se basant sur un solutionnaire ou en les comparant avec un petit ensemble de copies corrigées (entre 20 et 100).

IEA implémente aussi un tuteur automatique pour attirer l'attention des étudiants sur les points mal traités et aussi les rediriger vers les parties de cours qui y sont liées.

IEA attribue les notes selon trois critères de base : le contenu, le style de rédaction et le lexique.

Dans ses articles, Landauer ne divulgue pas comment ces différentes mesures sont évaluées dans le système (le IEA est la propriété de *Knowledge Analysis Technologies*) mais il rapporte quelques expériences [14]. Dans une de ces dernières, Landauer utilise 3296 copies de différents tests traitant 15 sujets différents. Les copies sont rédigées par des étudiants de différents niveaux d'études et différents domaines. Toutes les copies sont corrigées, de manière holistique, par deux correcteurs humains indépendants en plus de IEA. La moyenne des deux corrélations calculées avec chaque correcteur varie entre 0,73 et 0,85 sur des corpus où les deux correcteurs ont, entre eux, une corrélation de 0,75 et 0,86 respectivement.

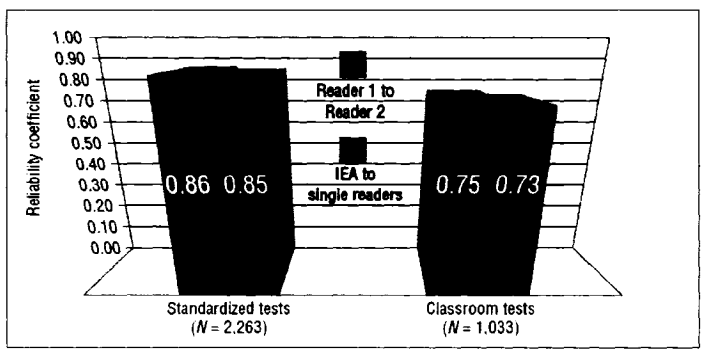

Figure 4.2 : Corrélation entre IEA et les correcteurs humains [14]

4.3 Évaluation de LSA

Dans la section précédente, nous avons introduit le domaine de la cotation automatique et nous avons présenté des exemples de systèmes de cotation. On a vu que ces systèmes évoluent d'une simple interprétation lexicale (par exemple PEG) vers une évaluation sémantique de textes (comme le IEA).

Dans cette section, nous présentons nos expériences d'évaluation de *LSA*. Cette technique utilise la décomposition en valeurs singulières (*SVD*) pour explorer les associations entre les mots. *LSA* permet de déduire le sens contextuel des mots et offre la possibilité d'une évaluation sémantique de textes. Cette capacité de déduction permet à *LSA* de mesurer la qualité des essais et ainsi d'approcher le raisonnement humain dans ce domaine.

Dans nos expériences, nous évaluons deux applications de *LSA*. La première est une catégorisation de documents selon les thèmes. La deuxième est une cotation automatique des essais.

La catégorisation de documents s'éloigne de l'objectif de cotation mais elle vise à démontrer l'efficacité de la technique *LSA* pour une tâche simple de classification. Cette expérience nous permet d'évaluer la sensibilité de *LSA* aux thèmes des textes et aussi d'explorer les problèmes de calibration du modèle.

La deuxième application est le sujet central de ce mémoire, c'est la cotation des essais. Cette application, plus délicate, est une évaluation de la capacité de *LSA* à évaluer la qualité des essais. Nous présentons différentes méthodes d'approcher le problème et à chaque test nous comparons les performances de *LSA* à ceux du *MEV*.

Avant d'étaler nos expériences, nous commençons par une brève introduction des outils principaux utilisés dans cette recherche.

4.3.1 Les outils

L'environnement R

Le logiciel *R* [23] est un environnement de manipulation de données basé sur le langage *S*. Conçu au départ pour la communauté scientifique, *R* offre des fonctionnalités avancées de calcul, des statistiques et des représentations graphiques. Entre autres, *R* permet une gestion efficace de grande quantité de données. Il prend en charge les calculs matriciels et il intègre une large collection d'outils d'analyse de données.

Nous utilisons *R* pour sa souplesse et sa puissance de programmation et aussi pour sa gratuité. Il est téléchargeable de son site officiel www.r-project.org avec un grand ensemble de bibliothèques et une bonne documentation.

Dans nos expériences, *R* est utilisé comme langage de programmation et comme calculateur. Grâce à *R*, le calcul de la *SVD* et la manipulation des matrices sont très simplifiés. En plus, les fonctions intégrées sont d'une grande utilité pour le calcul des statistiques sur les résultats.

Tree tagger

Tree tagger est un outil de traitement automatique de langue. Nous l'utilisons pour lemmatiser le texte des copies et le texte des articles qui constituent notre espace sémantique. Grâce à son interface en ligne de commande on peut facilement le combiner (en pipe) avec des commandes Linux pour enchaîner les traitements.

Tree tagger est un étiqueteur lemmatiseur adaptable à différentes langues dont l'espagnol, l'italien, le français, l'allemand et autres. Cet outil, développé à l'université de Stuttgart, atteint un taux de réussite de 94% [32] en plus d'être adaptable et facile à utiliser.

Les détails de ses caractéristiques techniques, la documentation et aussi la version complète sont disponibles sur le site de l'université de Stuttgart : www.ims.uni-stuttgart.de. On y trouve plusieurs extensions et fichiers paramètres pour chaque langue. Cet outil est disponible pour plusieurs plateformes dont Linux et Windows.

4.4 Classification avec LSA

Notre première expérience avec *LSA* est une simple classification de textes. Cette expérience nous permet de mettre au point la technique et de démontrer son efficacité. Plus précisément, nous nous intéressons à évaluer à quel point *LSA* est sensible au thème des textes.

Dans cette expérience, nous utilisons 120 essais d'étudiants traitant trois thèmes différents à raison de 40 copies par thème. Les trois thèmes sont nommés « *Gires* », « *Multiw* » et « *Sidoci* ». Nous retirons de chaque thème 10 essais. Cet ensemble de 30 copies, appelé « *témoins* », est utilisé pour tester l'acuité de la catégorisation. Le reste des 90 copies, appelé « *références* », sert à construire l'espace sémantique et, par la suite, à créer un vecteur moyen par thème. La structure du corpus est représentée dans la figure 4.3.

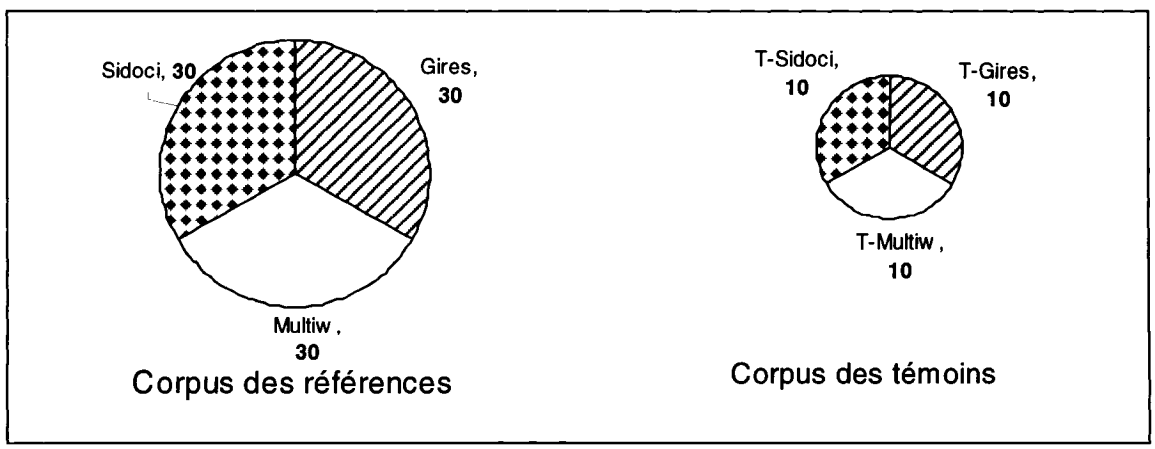

Figure 4.3 : Structure du corpus de classification

Pour classifier un témoin, nous le comparons à chaque vecteur moyen de thème puis nous l'affectons à la catégorie la plus proche. Nous testons ce même processus de classification pour des tailles différentes de l'espace sémantique. À chaque test, nous mesurons l'impact du nombre de dimensions sur la qualité de la classification.

Une catégorisation correcte, selon le thème, classifie les témoins en trois sous-groupes égaux de 10 copies chacun. Dans la suite nous représentons l'algorithme et les résultats de cette expérience.

4.4.1 Algorithme de la classification

Le processus de classification par *LSA* inclut les mêmes étapes utilisées par le *MEV* et présentées dans les sections précédentes (chapitre 2 et 3). Nous nous contentons, dans cette section, de les rappeler brièvement et de les appliquer à notre corpus d'essais.

La première étape consiste à adapter les documents. Puisque nos copies sont déjà en format numérique, cette tâche se limite à la suppression des caractères étrangers, noms des étudiants, matricules, etc.

La deuxième étape se résume à la suppression des mots vides (*Stop list*), et la troisième étape à la lemmatisation des copies, c'est-à-dire la recherche du « lemma » des mots (singulier masculin et verbe infinitif). Désormais, nous désignons ces racines par « *termes* ».

Dans la quatrième étape, nous représentons l'ensemble des copies sous forme d'une matrice *M* où la cellule *M[i,j]* représente le nombre d'occurrences du *terme i* dans le *document j*. La lemmatisation des 120 copies a généré 4176 *termes* c'est-à-dire que notre matrice des fréquences est composée de 4176 lignes et 120 colonnes. Chaque vecteur colonne représente un document.

La cinquième étape consiste à appliquer la pondération des termes pour refléter leurs poids dans la collection et dans le document. À partir de cette étape, nous traitons les fichiers témoins séparément des fichiers références pour que le poids des termes dans les témoins ne soit pas influencé par leur présence ou absence dans les fichiers références et vice versa.

Pour l'ensemble des fichiers références, nous utilisons la fonction *poids* présentée dans le chapitre 2. Cette fonction intègre deux composantes, une composante locale (au niveau de l'essai : *log(M[i,j]+1)*) et une composante globale (au niveau de corpus entier : *log(N/Df)+1)*). Dans le cas des documents références, la formule utilisée pour pondérer le *terme i* dans un *document j* est la suivante :

(Formule 4.1)
$$poids(i,j) = \frac{\log(M[i,j]+1) * (\log(N / Df_i)+1)}{\sqrt{\sum_{i=1}^{n} M[i,j]^2}}$$

où N représente le nombre de documents dans la collection des références (dans notre cas N= 90); n représente le nombre de termes dans la collection et Df_i le nombre de références où le *terme i* apparaît.

Pour la pondération des témoins, nous n'utilisons pas la composante globale pour une raison importante, nous voulons que la pondération des termes de chaque copie témoin soit indépendante de leur utilisation dans le reste des témoins. En plus, dans les applications réelles, contrairement aux expériences, on ne connaît pas a priori les caractéristiques de toutes copies que les étudiants pourront remettre. La formule 4.2 est utilisée pour pondérer un terme i dans un témoin j.

(Formule 4.2)
$$poids\_Temoin(i,j) = \frac{\log(M[i,j]+1)}{\sqrt{\sum_{i=1}^{n} M[i,j]^2}}$$

où *n* est le nombre de *termes* dans la collection.

Dorénavant, la matrice *M* fera référence à la matrice pondérée des essais références, tandis que la matrice des résultats de la pondération des témoins sera notée *MPT*.

À ce stade, les étapes présentées sont pareilles à ceux de l'approche *MEV*. La prochaine étape est la plus importante, elle démarque la *LSA* par rapport aux autres méthodes utilisant l'espace vectoriel. Il s'agit de la décomposition en valeur singulières (*SVD*) et de la projection des documents dans un espace réduit. Nous rappelons que cette technique permet de construire un espace réduit « *M'* » à *k* dimensions à partir de la décomposition en valeurs singulières de la matrice initiale *M* (formule 4.2). Le nouvel espace *M'* est donc sémantiquement plus précis que *M* grâce à l'élimination des dimensions peu significatives. Dans l'espace *M'*, on représente les documents références « *Doc* » par les vecteurs lignes de la matrice V_k, et les requêtes, « *Q* », ou les nouveaux documents, par la somme pondérée de leurs termes, (formule 4.3).

(Formule 4.2)
$$M'(n\times p) = U_k(n\times k) * diag(D_k)(k\times k) * t(V_k)(k\times p)$$
$$avec : k \le \min(n, p)$$

(Formule 4.3)
$$Q_k = t(Q) * U_k(n\times k) * diag(D_k^{-1})(k\times k)$$

Après la *SVD*, nous regroupons les essais références (*Ref*) par thème et nous représentons chaque groupe par la moyenne de ses essais, formule 4.4. Chacun des trois thèmes est ainsi représenté par un vecteur moyen, à savoir, « *VctMoyGires* », « *VctMoyMultiw* » et « *VctMoySidoci* ».

(Formule 4.4)
$$VctMoy = \frac{1}{30}\sum_{1}^{30} \mathrm{Ref}_k$$

La classification d'une copie témoin, *Temoin*, consiste à calculer sa similarité avec chacun des trois vecteurs moyens, (formule 4.5) et de la classer dans la catégorie la plus proche.

(Formule 4.5) $Sim = \cos(Temoin, VctMoy)$

La similarité varie entre -1 et 1 selon le « *cosinus* » de l'angle formé par les deux vecteurs à comparer. Plus la valeur du *cosinus* est grande, plus les deux vecteurs sont similaires.

4.4.2 Résultats de la classification :

4.4.2.1 Classification par MEV

Nous commençons par appliquer le *MEV* pour la tâche de classification des témoins. Ce modèle nous sert de référence pour évaluer l'efficacité de *LSA*. Nous rappelons que notre corpus référence se compose de 30 copies référence par thème (total de 90 copies), et que *M* est la matrice pondérée des fréquences de termes calculées à partir de ce corpus. *M* contient n=4176 lignes et p=90 colonnes où chaque colonne est les coordonnées d'un point dans un espace multidimensionnel de 4176 dimensions. Chacun de ces points représente un essai de notre corpus des références.

Nous commençons par calculer la moyenne de chaque thème dans notre corpus référence, formule 4.4. Ce point est considéré comme un vecteur dont la racine est l'origine du repère de l'espace. Ainsi nous obtenons les trois vecteurs précédemment présentés : *VctMoyGires*, *VctMoyMultiw* et *VctMoySidoci*. Nous basons la classification des témoins sur la similarité avec ces trois vecteurs.

La matrice *MPT*, la matrice pondérée des témoins, contient 4176 lignes et 30 colonnes. Chaque colonne est les coordonnées d'un point, essai témoin, dans l'espace à 4176 dimensions. Ainsi chaque *témoin* est représenté par un vecteur dont la racine est l'origine du repère. Nous rappelons que le corpus *témoins* se compose de 10 essais par thème pour un total de 30 témoins.

On évalue la similarité entre un témoin et un thème par la valeur du *cosinus* entre le vecteur du témoin et le vecteur moyen du thème, formule 4.5. Les résultats sont présentés dans le tableau 4.2. Les cellules grisées marquent les plus hauts niveaux de similarité.

***Tableau 4.2 : Valeurs des similarités entre les témoins et les vecteurs-thème selon le* MEV**

	Témoins	VctMoyGires	VctMoyMultiw	VctMoySidoci
Thème Gires	1	0.276	0.129	0.114
	2	0.173	0.120	0.128
	3	0.201	0.107	0.114
	4	0.188	0.120	0.130
	5	0.203	0.072	0.066
	6	0.157	0.114	0.098
	7	0.224	0.104	0.111
	8	0.188	0.132	0.107
	9	0.210	0.119	0.114
	10	0.226	0.121	0.111
Thème Multiw	11	0.099	0.208	0.102
	12	0.126	0.195	0.121
	13	0.110	0.194	0.111
	14	0.119	0.223	0.101
	15	0.118	0.207	0.121
	16	0.105	0.187	0.095
	17	0.100	0.178	0.086
	18	0.135	0.200	0.113
	19	0.125	0.239	0.118
	20	0.102	0.191	0.100

Tableau 4.2 : Valeurs des similarités entre les témoins et les vecteurs-thème selon le* MEV*(suite)

Thème Sidoci				
	21	0.101	0.108	0.197
	22	0.109	0.114	0.217
	23	0.121	0.124	0.229
	24	0.124	0.124	0.218
	25	0.117	0.126	0.219
	26	0.126	0.130	0.208
	27	0.116	0.124	0.195
	28	0.110	0.116	0.237
	29	0.116	0.126	0.187
	30	0.125	0.126	0.231

Le tableau des résultats montre que les vecteurs moyens ont les plus hauts nivaux de similarité avec les témoins du même thème qu'eux, c'est-à-dire que le *MEV* réussit à classifier correctement tous les témoins selon le thème, nous appelons ce critère le nombre de *témoins correctement catégorisés* (*TCC*). Pour mieux visualiser ce constat, nous représentons ces niveaux de similarité graphiquement dans la figure 4.4. Les valeurs possibles de la fonction *cosinus* sont entre [-1,1], cependant, l'intervalle [-1,0[n'est pas représenté sur la figure car toutes les valeurs de similarité calculées sont positives.

Comme il n'y a pas de similarité négative, nous avons représenté l'intervalle de similarité [0,1] au lieu de [-1,1] qui représente les valeurs possibles la fonction *cosinus*.

D'après la figure 4.4, on remarque que tous les fichiers témoins sont concentrés dans un petit intervalle de faible similarité, inférieur à 0,3.

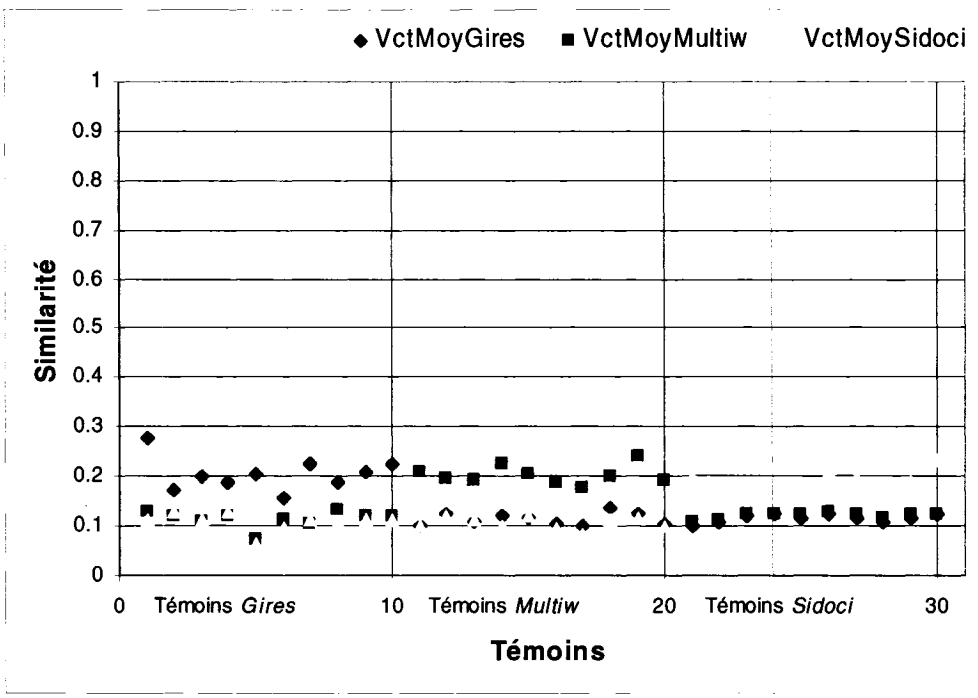

***Figure 4.4 : Niveaux de similarité avec le* MEV**

On remarque que les témoins ayant le plus haut niveau de similarité sont faiblement supérieurs aux autres témoins. Ces deux constatations nous ont guidé à définir deux critères pour mesurer la qualité de la classification.

Le premier critère, appelé « *Moyenne des similarités*» (*MS*), permet de répondre à la question : « sur une échelle de -1 à 1, quel est le niveau de similarité entre les fichiers témoins d'un thème et le vecteur moyen de leur catégorie ? ». Autrement dit, c'est une mesure du degré d'appartenance des témoins à leurs catégories. Puisque tous les témoins sont correctement classifiés, la valeur de *MS* correspond à la moyenne des cellules grisées de chaque thème sur le tableau 4.2. Pour un thème choisi, dont le vecteur moyen est *VctMoy*, nous calculons la moyenne des similarités en utilisant la formule 4.6.

(Formule 4.6) $$MS = \sum_{i=1}^{N} \cos(Temoin_i, VctMoy) / N$$

où *Temoin* et *VctMoy* sont du même thème et *N* le nombre de témoins par thème, dans notre cas *N*=10

Le tableau 4.3, présente les valeurs de la *MS* (cellules grisées) et aussi les valeurs des similarités entre les vecteurs moyens et les autres groupes de témoins.

***Tableau 4.3 : Moyenne de similarités par thème avec le* MEV**

	Témoins		
	Gires	Multiw	Sidoci
VctMoyGires	0,204	0,114	0,109
VctMoyMultiw	0,114	0,202	0,107
VctMoySidoci	0,116	0,122	0,214

Une valeur de *MS* proche de « 1 » indique que la méthode a détecté un niveau important de similarité entre les témoins et leur catégorie. Une valeur proche de « 0 » nous informe que la méthode n'a pas clairement établit le lien entre les témoins et leur catégorie et que cette dernière est choisie juste car elle est relativement plus proche que les deux autres choix. Une valeur proche de « -1 » signifie que la méthode a révélé un niveau important de disparité entre le contenu du témoin et celui de la catégorie et que le choix est porté sur la catégorie ayant le moins de points de dissimilitude (principe du « moindre mal »).

Les valeurs de *MS* sont évaluées à 0,2 ce qui signifie que le *MEV* réussit à détecter un faible niveau similarité entre le contenu du groupe des témoins et le vecteur moyen de leur catégorie.

Le deuxième critère, nommé "*discrimination*", nous informe sur le degré de précision de la technique. Nous voulons mesurer jusqu'à quel point la méthode de classification fait la distinction entre les groupes des témoins selon leurs thèmes. Ce critère est une mesure du niveau de certitude des résultats. Pour un thème, ayant le vecteur moyen *VctMoy* et la moyenne des similarités *MS*, nous calculons la discrimination en utilisant la formule 4.7.

(Formule 4.7) $$discrimination = MS - \frac{1}{N}\sum_{i=1}^{N} cos(Temoin_i, VctMoy)$$

où $Termoin_i$ et *VctMoy* traitent des thèmes différents et N représente le nombre des témoins n'appartenant pas au même thème que *VctMoy*, dans notre cas N=20.

La valeur de discrimination de chaque thème est calculée à partir de son vecteur moyen dans le tableau 4.3. Ceci correspond à la différence entre la valeur de la cellule grisée et la moyenne des deux autres cellules de la même ligne, le tableau 4.4 présente les résultats.

Tableau 4.4 : Niveau de discrimination par thème avec le MEV

	Témoins		
	Gires	Multiw	Sidoci
VctMoyGires	0,093	#	#
VctMoyMultiw	#	0,092	#
VctMoySidoci	#	#	0,095

La valeur de discrimination du *MEV* est très modeste (0.09), ce qui signifie que le modèle ne réussit pas à clairement distinguer le groupe de témoins correspondant à un vecteur moyen donné.

Malgré la justesse de la classification avec *MEV*, sa faible qualité (MS à 0,2 et discrimination à 0,09) nous laissent croire que dans un cas de catégorisation plus difficile, comme pour la cotation, ce modèle ne réussira pas à catégoriser correctement les copies. Nous allons maintenant évaluer *LSA* pour la même tâche de classification puis comparer les deux modèles.

4.4.2.2 Classification par LSA

L'analyse sémantique latente suit les mêmes étapes que le *MEV* en plus d'intégrer la décomposition en valeurs singulières. Nous allons évaluer *LSA* pour une tâche de classification de textes selon leurs thèmes. Nous disposons de trois critères d'évaluation, le nombre de *témoins correctement classifiés* (TCC) qui mesure l'efficacité de la classification, la *moyenne des similarités* pour mesurer le degré d'appartenance des témoins à leur catégorie, et le troisième critère, nommé *discrimination*, pour évaluer le niveau de certitude des résultats.

Pour cette expérience, nous avons réduit l'espace sémantique sur différent nombre de dimensions. Ce nombre, noté k, varie entre 2 et 90 (formule 4.2). Pour chaque taille de l'espace réduit, nous appliquons l'algorithme de classification et nous évaluons la qualité des résultats.

La matrice M est une matrice de n=4176 lignes et p=90 colonnes, représentant les fréquences pondérées des *termes* dans le corpus des références. Cette matrice est un espace de 4176 dimensions où chacun des 90 documents est représenté par un point. Nous appliquons la décomposition en valeurs singulières pour décomposer la matrice M en trois facteurs : deux matrices, U et V, et le vecteur D. Les valeurs du vecteur D sont triées en ordre décroissant et reflètent le poids de chaque dimension dans l'espace. Par exemple, la projection de cet espace sur ses 10 dimensions les plus significatives revient à mettre à zéro toutes les valeurs de D sauf les 10 premières, puis de calculer le nouvel espace M', la formule 4.2, renommée ici formule 4.7, rappelle l'expression mathématique générale pour une projection sur k dimensions.

(Formule 4.8)[3]

$$M'(n \times p) = U_k(n \times k) * diag(D_k)(k \times k) * t(V_k)(k \times p)$$
$$avec : k \leq \min(n, p)$$

Les lignes de U_k et V_k sont respectivement les vecteurs des *termes* et des documents références dans l'espace réduit à k dimensions.

Notre corpus référence se compose de 90 essais références (30 essais par thème), nous représentons chaque thème par la moyenne de ses références, nous nommons ses vecteurs, $VctMoyGires_K$, et $VctMoyMultiw_K$ et $VctMoySidoci_K$, où k est le nombre de dimensions de l'espace réduit.

Nous utilisons la matrice des fréquences pondérées de témoins *MTP* pour calculer la similarité des témoins avec chaque thème. Il faut donc représenter les vecteurs des témoins, dans l'espace réduit *M'* à k dimensions, pour cela nous utilisons la formule 4.8

(Formule 4.9) [3]

$$temoin_k = t(temoin) * U_k(n \times k) * diag(D_k^{-1})(k \times k)$$

où *t(temoin)* est la transposée du vecteur *temoin.*

Par la suite, nous calculons les similarités et les mesures d'évaluation de la classification par *LSA* de même façon que précédemment pour le *MEV.*

Guidé par les critères d'évaluations, nous allons explorer les différentes tailles de l'espace réduit jusqu'à trouver les meilleurs résultats. Nous commençons par un espace à deux dimensions, c'est-à-dire $k=2$. Dans ce cas, toutes les copies témoins sont représentées sur les deux dimensions les plus significatives de l'espace. Le tableau 4.5 présente les valeurs de similarité entre chaque vecteur thème et ensemble des témoins, les plus grandes valeurs y sont grisées.

Tableau 4.5 : Similarités entre les témoins et les vecteurs thème avec* LSA *(k=2)

	Témoins	$VctMoyGires_2$	$VctMoyMultiw_2$	$VctMoySidoci_2$
Thème Gires	1	1.000	0.826	0.870
	2	0.999	0.799	0.892

	3	0.998	0.788	0.900
	4	0.997	0.771	0.911
	5	1.000	0.832	0.865
	6	0.991	0.892	0.800
	7	0.999	0.802	0.890
	8	0.999	0.847	0.850
	9	1.000	0.827	0.869
	10	1.000	0.829	0.867
Thème Multiw	11	0.966	0.941	0.719
	12	0.984	0.909	0.774
	13	0.950	0.959	0.678
	14	0.906	0.985	0.587
	15	0.952	0.956	0.684
	16	0.960	0.949	0.703
	17	0.951	0.958	0.681
	18	0.977	0.924	0.750
	19	0.964	0.943	0.715
	20	0.967	0.939	0.722
Thème Sidoci	21	0.945	0.590	0.985
	22	0.929	0.552	0.992
	23	0.948	0.597	0.983
	24	0.957	0.622	0.977
	25	0.959	0.625	0.976
	26	0.963	0.638	0.973
	27	0.974	0.670	0.962
	28	0.935	0.566	0.990
	29	0.965	0.644	0.971
	30	0.953	0.611	0.980

La première remarque est que *LSA* (k=2), ne réussit pas à classifier correctement quelques témoins. Ainsi, le nombre de témoins correctement classifiés (*TCC*) n'est pas à son niveau optimal (Tableau 4.6). On remarque aussi que les valeurs de similarité sont beaucoup plus élevées que dans le cas du *MEV*. La figure 4.4 représente graphiquement les valeurs calculées.

Tableau 4.6 : Nombre de témoins correctement classifiés avec* LSA *(k=2)

	Corpus de témoins			Total
	Gires	Multiw	Sidoci	
TCC	10	4	9	23

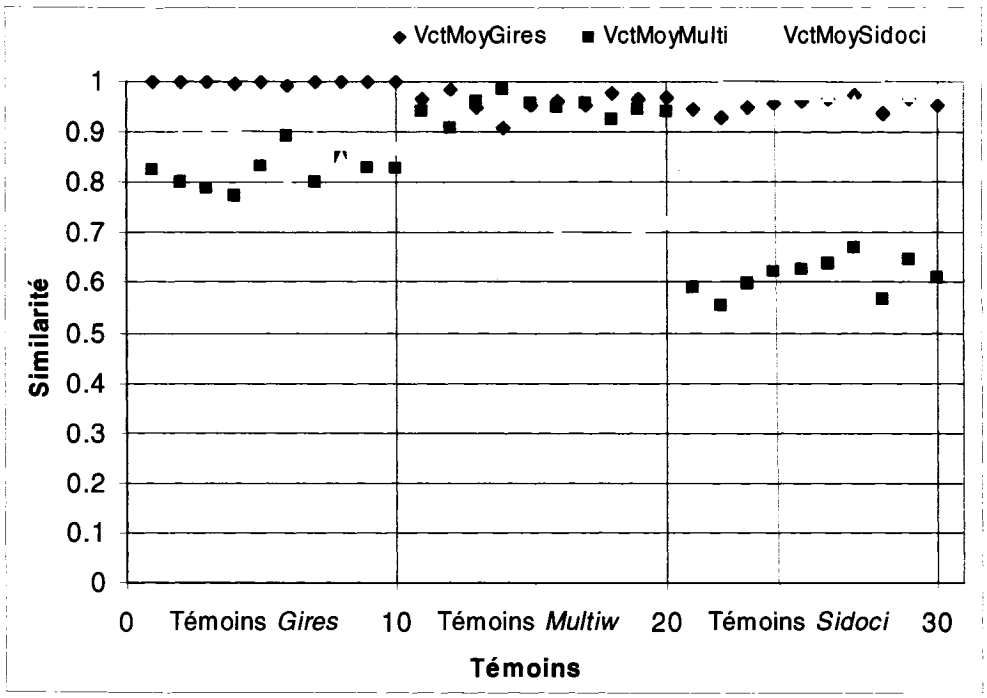

Figure 4.5 : Niveaux de similarité avec* LSA *(k=2)

La figure 4.4 confirme le haut niveau de similarité moyenne et affiche une amélioration significative dans la discrimination des catégories par rapport au *MEV*. Le deuxième critère, la moyenne des similarités, mettra en évidence cette haute similarité. (Tableau 4.6)

Tableau 4.7 : Moyenne de similarité par thème avec LSA (k=2)

	Témoins		
	Gires	Multiw	Sidoci
$VctMoyGires_K$	0.998	0.821	0.871
$VctMoyMultiw_K$	0.957	0.946	0.701
$VctMoySidoci_K$	0.952	0.611	0.978

À partir du tableau des moyennes de similarité, nous calculons le niveau de discrimination, formule 4.7. Les résultats sont présentés dans le tableau 4.8.

Tableau 4.8 : Niveau de discrimination par thème avec* LSA *(k=2)

	Témoins		
	Gires	Multiw	Sidoci
$VctMoyGires_K$	0.151	#	#
$VctMoyMultiw_K$	#	0.116	#
$VctMoySidoci_K$	#	#	0.196

En considérant deux dimensions seulement, *LSA* trouve que les fichiers témoins sont très similaires (SM≈0.9), ce qui signifie que les deux dimensions retenues sont présentes dans toutes les copies. D'un autre côté, la faible discrimination (≈0.1) montre que, pour k=2, *LSA* ne détecte pas une grande différence entre les catégories ce qui se reflète dans le score au TCC (23 sur 30).

Les résultats de la classification avec *LSA* (k=2) sont assez modestes, ceci est lié au petit nombre de dimensions considérées. En réduisant ce nombre à deux, *LSA* ne dispose pas d'assez d'information pour estimer efficacement les liens entre les témoins et les catégories.

Pour le cas d'un espace réduit à deux dimensions, on peut représenter graphiquement les positions des copies témoins sur un plan. La figure 4.6 montre que ces dernières sont assez bien séparées en trois groupes selon leur thème. Le faible taux de discrimination se manifeste par le petit intervalle de distribution des copies aussi bien sur l'axe des abscisses (0,005 à 0,03) que sur l'axe des ordonnées (-0,01 à 0,01)

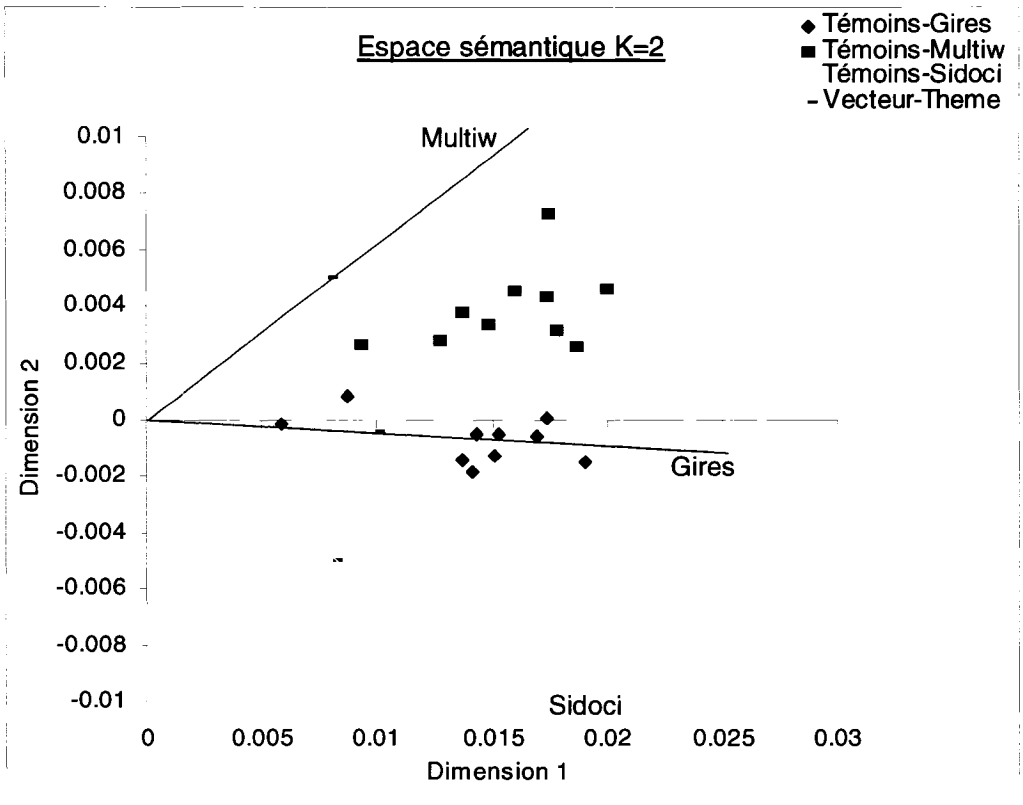

Figure 4.6 : Positions des témoins dans l'espace sémantique à deux dimensions

Nous continuons à explorer les différents nombres de dimensions. À l'étape suivante nous évaluons un espace à trois dimensions.

Tableau 4.9 : Similarités entre les témoins et les vecteurs thème avec* LSA *(k=3)

	Témoins	$VctMoyGires_3$	$VctMoyMultiw_3$	$VctMoySidoci_3$
Thème Gires	1	0.996	0.745	0.098
	2	0.888	0.805	0.453
	3	0.955	0.772	0.303
	4	0.898	0.777	0.446
	5	0.999	0.730	0.048
	6	0.921	0.880	0.309
	7	0.971	0.771	0.246
	8	0.975	0.805	0.203
	9	0.966	0.798	0.250
	10	0.981	0.782	0.191

Tableau 4.9 : Similarités entre les témoins et les vecteurs thème avec* LSA *(k=3) (suite)

	Témoins	$VctMoyGires_3$	$VctMoyMultiw_3$	$VctMoySidoci_3$
Thème Multiw	11	0.807	0.943	0.415
	12	0.831	0.913	0.441
	13	0.769	0.959	0.418
	14	0.808	0.983	0.258
	15	0.759	0.955	0.436
	16	0.871	0.943	0.300
	17	0.851	0.955	0.306
	18	0.892	0.918	0.316
	19	0.822	0.945	0.389
	20	0.844	0.940	0.366
Thème Sidoci	21	0.160	0.374	0.984
	22	0.198	0.366	0.982
	23	0.225	0.411	0.974
	24	0.291	0.461	0.956
	25	0.294	0.465	0.955
	26	0.315	0.485	0.947
	27	0.338	0.522	0.935
	28	0.154	0.354	0.987
	29	0.349	0.506	0.936
	30	0.250	0.433	0.967

LSA (k=3) réussit une catégorisation correcte pour l'ensemble des 30 copies témoins. Quant au niveau de discrimination, ce dernier s'améliore considérablement (tableau 4.9 et tableau 4.10). Ces améliorations sont clairement visibles sur la figure 4.3.

Tableau 4.10 : Moyenne de similarité par thème avec* LSA *(k=3)

	Témoins		
	Gires	Multiw	Sidoci
$VctMoyGires_3$	0.955	0.786	0.254
$VctMoyMultiw_3$	0.825	0.945	0.364
$VctMoySidoci_3$	0.257	0.437	0.962

Tableau 4.11 : Niveau de discrimination par thème avec* LSA *(k=3)

	Témoins		
	Gires	Multiw	Sidoci
$VctMoyGires_3$	0.434	#	#
$VctMoyMultiw_3$	#	0.350	#
$VctMoySidoci_3$	#	#	0.614

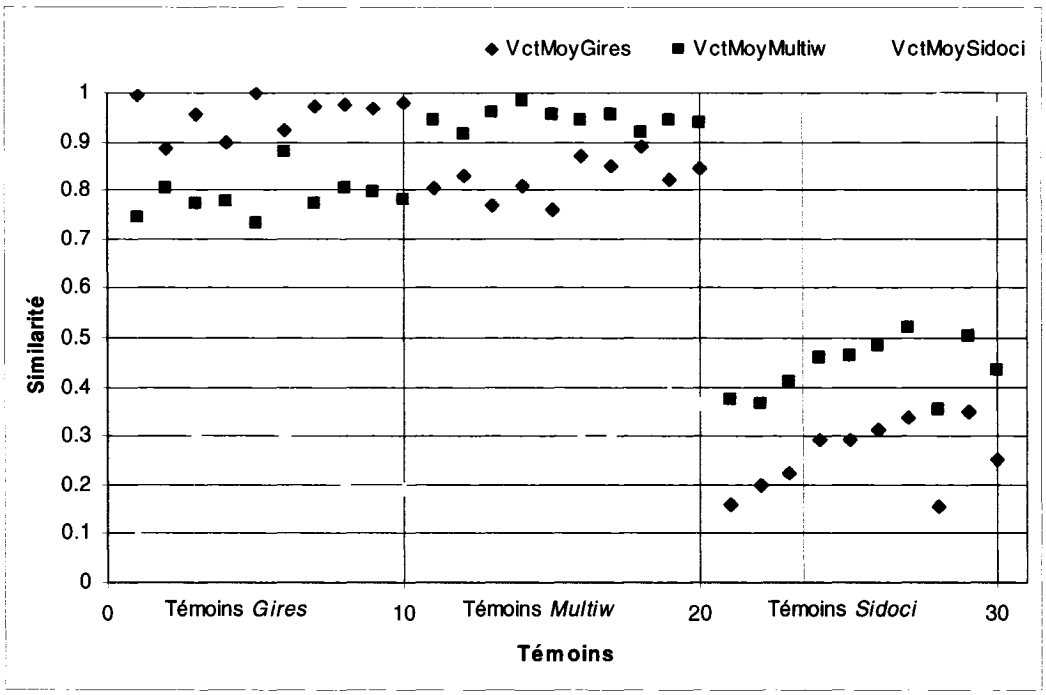

Figure 4.7 : Niveaux de similarité avec* LSA *(k=3)

Pour k=3, on remarque une nette amélioration du niveau de discrimination relativement à k=2. On constate cependant que les témoins « Gires » et « Multiw » sont assez proches et que la distinction entre eux n'est pas encore très prononcée.

Les meilleurs résultats sont constatés sur un espace réduit à six dimensions, k=6. D'un coté, on y obtient le meilleur taux de discrimination (≈0,74) ce qui signifie que *LSA* peut facilement juger de la catégorie de chaque copie. De l'autre coté, on remarque que la moyenne de similarités est très prononcée (≈0,95) ce qui confirme que *LSA* (*k*=6) détecte aisément la similarité du contenu entre les témoins et le vecteur moyen de leur catégorie.

Les tableaux 4.11, 4.12 et 4.13 présentes les valeurs de nos mesures de la qualité de la classification. Aussi la figure 4.8 met en évidence la nette amélioration de ces critères.

Tableau 4.12 : Similarités entre les témoins et les vecteurs thème avec* LSA *(k=6)

	Témoins	$VctMoyGires_6$	$VctMoyMultiw_6$	$VctMoySidoci_6$
Thème Gires	1	0.983	0.059	0.101
	2	0.875	0.266	0.408
	3	0.961	0.086	0.271
	4	0.893	0.217	0.412
	5	0.994	0.004	0.069
	6	0.879	0.369	0.271
	7	0.969	0.059	0.229
	8	0.925	0.346	0.184
	9	0.965	0.185	0.245
	10	0.971	0.197	0.186
Thème Multiw	11	0.044	0.972	0.183
	12	0.220	0.936	0.262
	13	0.143	0.970	0.246
	14	0.116	0.987	0.100
	15	0.166	0.962	0.257
	16	0.171	0.966	0.141
	17	0.155	0.973	0.139
	18	0.333	0.921	0.189
	19	0.140	0.967	0.205
	20	0.129	0.967	0.172

Tableau 4.12 : Similarités entre les témoins et les vecteurs thème avec* LSA *(k=6) (Suite)

Thème Sidoci	21	0.159	0.218	0.971
	22	0.191	0.209	0.974
	23	0.218	0.215	0.971
	24	0.284	0.261	0.946
	25	0.228	0.323	0.944
	26	0.270	0.312	0.940
	27	0.273	0.356	0.919
	28	0.157	0.193	0.984
	29	0.249	0.375	0.924
	30	0.227	0.230	0.958

Tableau 4.13 : Moyenne de similarité par thème avec* LSA *(k=6)

	Témoins		
	Gires	Multiw	Sidoci
$VctMoyGires_6$	**0.941**	0.178	0.237
$VctMoyMultiw_6$	0.161	**0.962**	0.189
$VctMoySidoci_6$	0.225	0.269	**0.953**

Tableau 4.14 : Niveau de discrimination par thème avec LSA (k=6)

	Témoins		
	Gires	Multiw	Sidoci
$VctMoyGires_6$	**0.733**	#	#
$VctMoyMultiw_6$	#	**0.786**	#
$VctMoySidoci_6$	#	#	**0.705**

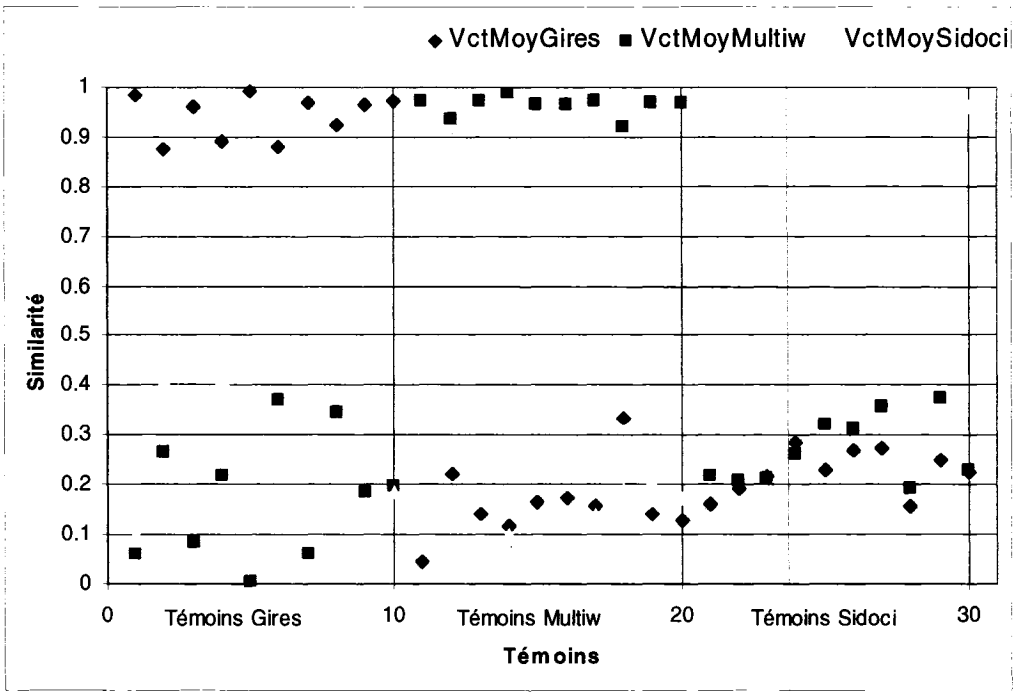

Figure 4.8 : Niveau de similarité avec le* LSA *(k=6)

Pour les espaces sémantiques réduits sur un nombre de dimensions supérieur à 6, *LSA* réussit toujours une catégorisation correcte mais le niveau de discrimination se détériore graduellement jusqu'à atteindre une valeur proche de 0,55 avec un k=90 (figure 4.9).

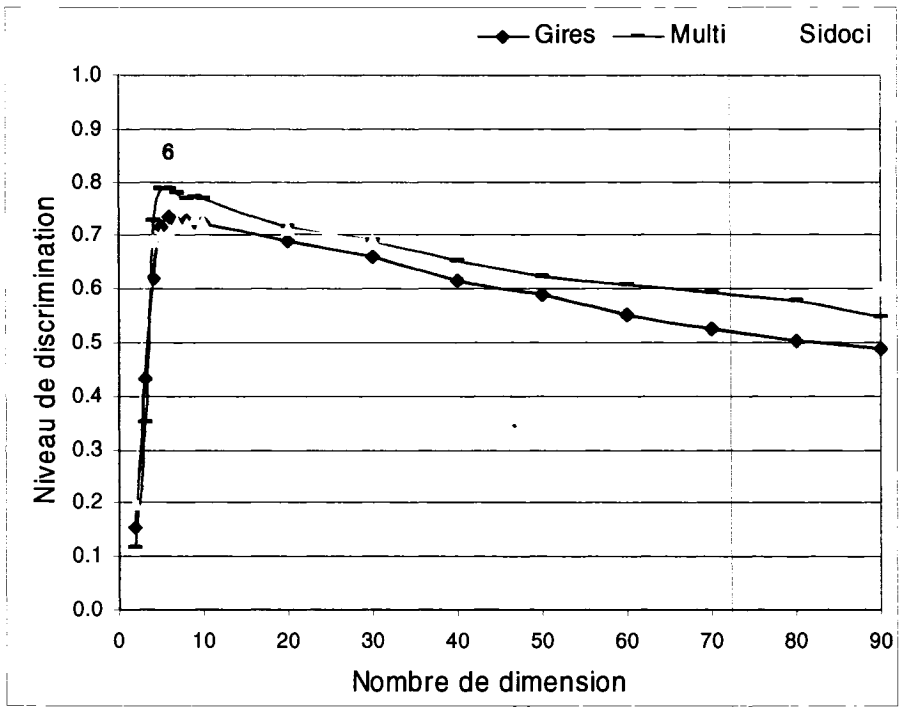

***Figure 4.9 : Évolution de la discrimination selon le nombre de dimensions avec* LSA**

Cette expérience confirme que *LSA* est plus précise que le *MEV* pour une application de classifications de textes. Ce résultat est atteint grâce à la réduction de l'espace initial qui extrait les liens sémantiques entre les thèmes des témoins et leur catégorie correspondante.

D'un autre coté, cette expérience nous a permis d'évaluer l'impact du nombre de dimensions (k) sur l'efficacité de *LSA*. Dans les tests rapportés dans la littérature [2, 3,4], *LSA* est jugée plus performante sur des espaces sémantiques réduits ayant entre 50 et 400 dimensions. Ceci est vrai pour des corpus de grande taille (environ 30 000 articles et près de 11 millions mots). Dans notre cas, les meilleurs résultats sont obtenus avec un espace réduit de 6 dimensions, ce qui est proportionnel à la taille de notre corpus (90 copies et 4176 mots).

À ce stade, nous avons illustré la capacité de *LSA* à détecter le contenu des textes. Dans la section suivante, nous évaluons cette technique pour une application, plus délicate, qui consiste à juger de la qualité des essais.

4.5 La cotation avec LSA

Le but principal de ce mémoire est d'évaluer la technique d'analyse sémantique latente pour la cotation des essais. Nous avons commencé par une simple catégorisation de textes pour illustrer le fonctionnement de cette technique et confirmer son efficacité et sa capacité à détecter le contenu de textes. La catégorisation de textes nous a permis aussi d'explorer l'espace des paramètres qui influencent la performance de *LSA* et nous pouvons maintenant entamer la tâche de cotation qui est plus exigeante que celle de la classification des cas.

La cotation des essais a pour but d'apprécier la qualité de textes. Les propriétés lexicales sont jugées insuffisantes pour une évaluation qualitative. L'objectif derrière l'utilisation de *LSA* est d'explorer les associations latentes entre les termes pour évaluer la valeur sémantique du texte et simuler ainsi le raisonnement humain pour une cotation sémantique.

Dans cette section, nous présentons nos expériences de cotation d'essais avec *LSA* et nous comparons les notes attribuées par cette dernière avec celles du modèle *MEV*. La comparaison consiste à calculer le niveau de corrélation de chaque modèle avec la moyenne des correcteurs humains. Cette corrélation est calculée avec la formule de *Pearson*, formule 4.9.

(Formule 4.9)
$$pearson(x,y) = \frac{1}{n} \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\frac{1}{n}\sum_{i=1}^{n} (x_i - \bar{x})^2} * \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \bar{y})^2}}$$

où x et y sont deux vecteurs de documents et n le nombre de dimensions de l'espace.

Pour cette application de cotation, nous disposons de 77 copies d'une étude de cas reliée à la gestion de projet. Toutes les copies répondent au même devoir de quatre questions sur environ deux pages de texte. L'ensemble des 77 copies été corrigés par deux correcteurs humains. Nous avons divisé le corpus en trois sous-corpus : le premier appelé « *corpus référence* » contient 26 copies, le deuxième appelé « *corpus d'entraînement* » comporte 25 copies et le dernier, appelé « *corpus de validation* », regroupe les 26 copies restantes, la figure 4.10 présente la structure du corpus. Dans la section suivante, nous présentons notre algorithme générique de la cotation et nous expliquons la manière dont nous utilisons ces trois sous-corpus.

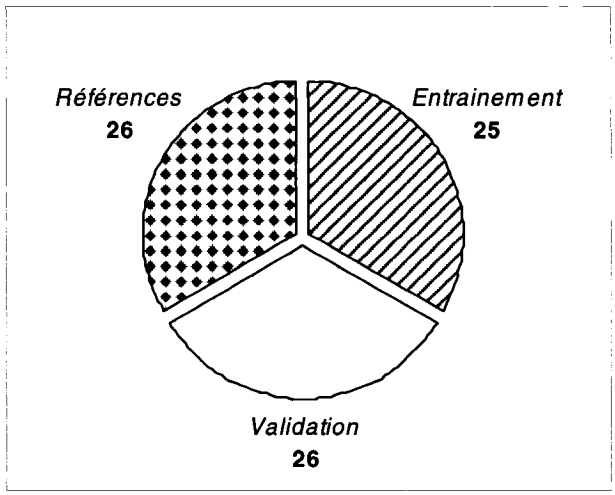

Figure 4.10 : Structure du corpus de la cotation

4.5.1 Algorithme de cotation

Nous avons abordé le problème de cotation avec deux approches différentes. Pour chaque approche nous avons testé deux techniques. Les étapes décrites dans cette section sont communes à toutes les méthodes de cotation testées. Nous présentons les détails spécifiques de chaque méthode plus loin dans ce chapitre.

La première étape notre algorithme consiste à créer un espace sémantique où toutes les copies vont être représentées. Cet espace représente les connaissances de *LSA* dans le domaine de l'évaluation. Ce dernier est généralement créé à partir de grandes collections et de livres de référence du domaine en question. Dans notre première expérience, nous avons exploré l'impact de la quantité et le contenu de cette collection sur la qualité de la cotation. Nous avons évalué deux collections, la première, qualifiée de « *connexe* », contient seulement des documents liés au domaine de la gestion de projet. La deuxième collection, dite « *générale* », est plus riche. Elle contient des documents généraux en plus des documents connexes. Nous exposons les résultats dans la section de la première expérience.

La deuxième étape concerne la préparation des copies. Nous suivons les mêmes étapes décrites précédemment pour la catégorisation de textes. Nous débutons par "nettoyer", lemmatiser et représenter les copies dans une matrice de fréquences. Par la suite, nous générons trois matrices selon les trois sous-corpus, une matrice de copies de référence notées *MRef*, une matrice d'entraînement notée *MEnt* et une matrice de validation notée *MVal*. En suite, nous appliquons la pondération de chaque matrice à part, ainsi on évite les interférences entre les poids des termes dans chaque corpus.

La troisième étape dans notre processus de cotation consiste à représenter les copies dans l'espace sémantique et de les coter selon leurs positions. Les trois sous-corpus sont utilisés de la même façon tout au long de nos expériences : nous considérons les copies du *corpus de référence* comme exemples de cotation. Elles nous servent de repère pour juger de la qualité des autres copies. La qualité d'une copie est estimée selon la similarité de son contenu avec les copies de *référence*. C'est à ce niveau que nous utilisons *LSA* pour juger de la proximité sémantique entre les copies. Nous profitons ainsi de la capacité de *LSA* à détecter les liens sémantiques entre les termes pour approcher le plus possible la cotation humaine.

Nous utilisons les copies du corpus d'entraînement pour calibrer le modèle de cotation et définir les paramètres optimaux. Nous attribuons des notes aux copies d'entraînement et nous ajustons nos paramètres de cotation (nombre de dimensions de l'espace et le nombre de copies de référence à considérer) pour approcher le mieux possible les notes attribuées par les correcteurs humains pour ce même corpus.

La calibration du modèle consiste à réduire l'espace sémantique sur un nombre différent de dimensions et pour chaque espace réduit nous explorons les paramètres de la méthode de cotation. Une fois notre modèle ajusté, nous fixons ces paramètres puis nous l'utilisons pour coter les copies du corpus de validation. On évite ainsi de tester le modèle avec les mêmes copies utilisées pour la calibration.

À chaque fois que nous attribuons des notes aux copies de validation, nous les comparons à ceux de la cotation avec le *MEV*. Nous utilisons la moyenne des notes des correcteurs humains comme critère pour comparer les deux modèles. Dans la section suivante, nous présentons les approches et les techniques utilisées pour la cotation.

4.5.2 Approches de cotation

Le devoir d'étude de cas se compose de quatre questions indépendantes. Puisque chaque question traite un aspect spécifique, nous avons décidé d'aborder le problème de cotation avec deux approches différentes. Dans la première, on considère les copies en entier comme une seule entité de texte, c'est ce qu'on appelle une « cotation holistique ». La cotation holistique évalue la copie entière pour l'ensemble des points traités concernant les quatre questions. Dans la deuxième approche, appelée « cotation modulaire », on corrige chaque réponse à part et on affecte la somme des notes à la copie.

Pour chacune des deux approches, nous avons testé deux techniques de cotation, une cotation selon le principe des « voisins les plus proches » et une cotation par « classification ». En utilisant la première technique, chaque copie reçoit la somme pondérée des notes de ses voisins références les plus proches. La qualité de la copie est ainsi évaluée selon ses voisins. Avec la deuxième technique, nous utilisons le même principe que la catégorisation : on divise les copies références en trois classes selon la qualité (les bonnes, les moyennes et les mauvaises) puis on représente chaque niveau par la moyenne de ses copies. Ainsi, la copie à corriger reçoit la cote du niveau le plus similaire.

Dans la section des résultats, nous appliquons une technique à la fois puis nous la comparons au *MEV* et aux correcteurs humains.

4.5.3 Cotation holistique

Une cotation holistique s'intéresse au contenu général des copies. Elle sert à évaluer les connaissances de l'étudiant sans l'obliger à se conformer à une structure précise. Un bon exemple est la dissertation libre où l'étudiant a une grande liberté dans l'organisation de ses idées et la présentation des réponses. On ne s'intéresse pas à la présentation mais plutôt au contenu.

4.5.3.1 Cotation holistique selon les voisins les plus proches

L'idée derrière la cotation holistique selon les voisins est que, dans un espace sémantique, les documents les plus similaires seront rapprochés. Ainsi deux copies utilisant les mêmes arguments et les mêmes idées auront des représentations rapprochées et devraient recevoir la même note. Si on représente dans cet espace un ensemble de copies de qualité connue (corpus *référence*) alors on peut mesurer la qualité de toute autre copie (corpus *d'entraînement* et de *validation*) selon ses voisins les plus proches parmi les copies références.

Dans cette expérience, nous avons testé deux types d'espace sémantique appelés *espace sémantique connexe* et *espace sémantique général*. Le premier est construit à partir de documents connexes au domaine de la gestion de projet, alors que le deuxième espace est composé d'un corpus général incluant des romans, des poèmes et aussi des articles de différents sujets dont la gestion de projet. Le but est de mesurer l'impact de la diversité de la matrice sémantique sur les résultats de la cotation.

Pour chaque espace considéré, il faut définir deux paramètres importants : le nombre de voisins à considérer, noté « h », et le nombre de dimensions optimal de l'espace réduit, noté « k ».

Notre corpus de référence contient 26 copies seulement. Nous avons donc estimé que notre premier paramètre, le nombre maximal de voisins considérés pour chaque note, ne doit pas dépasser 10, sinon les notes attribuées se rapprochent et tendent vers la moyenne du corpus référence.

Le deuxième paramètre est le nombre optimal de dimensions de l'espace réduit. À l'instar de ce que nous avons réalisé pour la tâche de classification de textes (figure 4.9) nous déterminons cette valeur en parcourant l'intervalle des valeurs possibles. Pour chaque valeur testée, nous attribuons des notes aux copies d'entraînement et nous les comparons avec la moyenne des notes des correcteurs humains et nous retenons les paramètres de la meilleure performance.

Espace sémantique connexe

L'espace sémantique connexe se compose d'un nombre réduit de documents au sujet de la gestion de projet. Ces documents sont des devoirs traitant d'autres sujets que celui utilisé dans cette expérience. Nous utilisons cet ensemble de 79 documents pour construire la matrice initiale.

La matrice initiale, nommée M, est composée de n=2971 lignes et p=79 colonnes. L'espace réduit aura donc entre 2 et 79 dimensions ($k \leq \min(n,p)$). Nous commençons par calculer la décomposition en valeurs singulières de la matrice M, puis nous explorons les différents espaces réduits en utilisons la formule 4.10 (citée précédemment formule 4.2). À chaque fois que nous générons un espace, nous y représentons les copies références et les copies d'entraînement, formule 4.11.

(Formule 4.10)

$$M'(n \times p) = U_k(n \times k) * diag(D_k)(k \times k) * t(V_k)(k \times p)$$
$$avec : k \leq \min(n, p)$$

(Formule 4.11)

$$Doc_k = t(Doc) * U_k(n \times k) * diag(D_k^{-1})(k \times k)$$

Par la suite, nous fixons une valeur pour le nombre de voisins à considérer, noté *h*. Pour chaque copie d'entraînement, noté *Ent*, nous calculons son niveau de similarité avec toutes les copies références, noté *Ref*, puis nous retenons les *h* meilleurs niveaux de similarité. La copie reçoit donc la note pondérée de ces *h* voisins les plus proches parmi les références, formule 4.12.

(Formule 4.11)

$$note\_Ent = \frac{\sum_{i=1}^{h} [Sim(Ent, ref_i) * note\_ref_i]}{\sum_{i=1}^{h} Sim(Ent, ref_i)}$$

où ref_i sont les références les plus similaires à la copie *Ent*.

Enfin nous calculons la corrélation entre *LSA* et les correcteurs humains. La figure 4.11 présente les meilleurs résultats pour les copies d'entraînement.

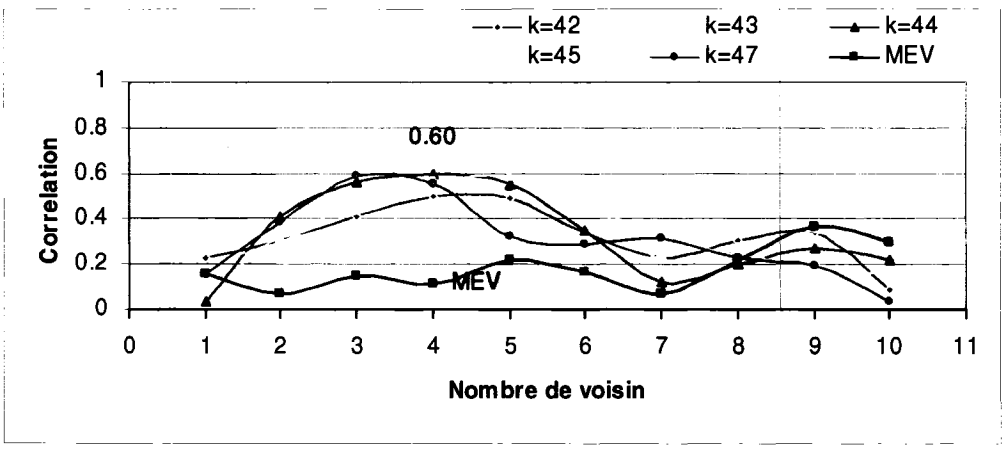

Figure 4.11 : Niveaux de corrélation pour les copies d'entraînement selon le nombre de dimensions de l'espace connexe réduit.

Nous remarquons que les meilleurs résultats résident dans un espace de 42 à 47 dimensions en utilisant 3 à 5 voisins les plus proches. Le maximum obtenu est une corrélation de 0,60 sur un espace à 44 dimensions et ceci en utilisant les quatre voisins les plus proches.

Nous fixons les valeurs de la meilleure performance, puis nous appliquons le modèle ajusté pour la cotation des copies de *validation.* Nous avons trouvé une corrélation de 0,36 entre les notes attribuées par *LSA* et la moyenne de ceux attribuées par les correcteurs humains. Cette corrélation est supérieure au taux de 0,23 obtenu entre ces derniers et le *MEV.* Notons ici que la correction entre les deux correcteurs humains est de 0,43. Le tableau 4.6 présente les notes attribuées par *LSA*, *MEV* et la moyenne des notes des correcteurs humains pour le corpus de validation.

Tableau 4.15 : Cotation du corpus de validation sur un espace sémantique connexe

Copies de validation	Moyenne des deux correcteurs	*LSA*	*MEV*
1	15,5	16,33	15,13
2	16,75	14,01	16,14
3	14,75	15,00	15,63
4	13,75	13,35	15,64
5	17,5	12,49	15,24
6	15	14,32	14,01

7	15,5	13,16	11,99
8	16,5	14,00	15,64
9	15,75	14,68	14,40
10	15,5	14,81	16,13
11	12,75	13,50	15,88
12	17,5	16,67	14,93
13	14,75	12,84	15,74
14	18,25	12,32	15,49
15	10,75	13,17	14,63
16	16,75	14,33	15,00
17	15,25	11,83	15,13
18	15	14,67	15,65
19	16	16,83	15,38
20	14,75	13,00	14,39

Tableau 4.15 : Cotation du corpus de validation sur un espace sémantique connexe (Suite)

21	13,25	11,66	13,47
22	14	14,66	15,49
23	18,5	16,50	16,49
24	13,25	13,82	15,75
25	17,5	14,84	15,13
26	14	12,33	12,89
Corrélation avec les correcteurs :		0,36	0,23

Espace sémantique général

Le second espace sémantique utilisé est plus large, *l'espace sémantique général.* Nous utilisons un grand corpus de documents français en provenance du site de l'Association des Bibliophiles Universels « abu.cnam.fr. ». Parmi les documents utilisés on cite « Notre dame de paris » de V. Hugo, « Les trois mousquetaires » de Dumas, « la vie sur Mars », « L'origine des espèces », « La théorie physique, son objet et sa structure », en plus des articles de presse concernant la gestion de projet et bien d'autres documents. L'ensemble du corpus compte 362 documents totalisant près deux millions (1.9M) de mots dont 9216 *termes.*

À l'exemple de l'espace connexe, nous explorons l'espace sémantique général sur un nombre de dimensions varié, et à chaque dimension, nous évaluons l'impact du nombre de voisins considérés sur le niveau de la corrélation avec les correcteurs humains. La figure 4.12 montre l'évolution du niveau de corrélation pour les cas les plus pertinents.

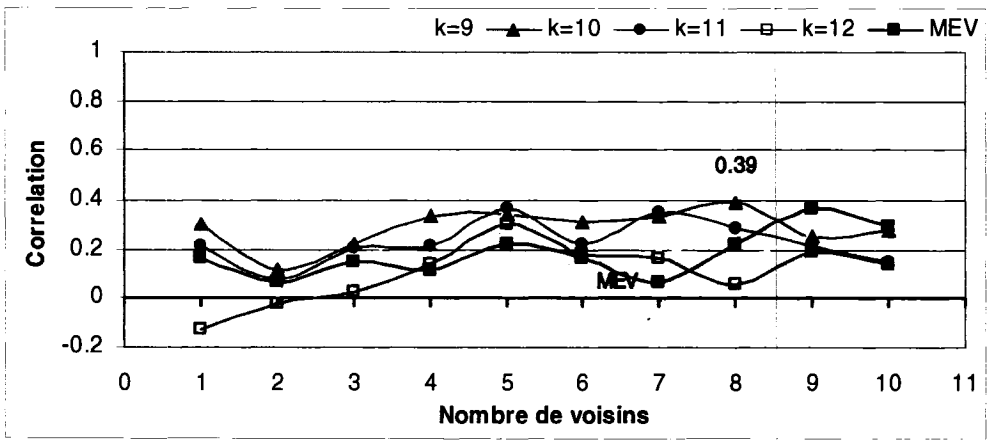

Figure 4.12 : Niveau de corrélation des notes d'entraînement selon le nombre de voisins

On remarque que les meilleurs paramètres sont l'espace à 10 dimensions avec l'utilisation de 8 copies voisines les plus proches. À ce point, *LSA* a un niveau de corrélation de 0,39 alors que celui de *MEV* est à 0,21.

On adopte ces mêmes valeurs de paramètres pour noter les copies de validation. On obtient un niveau de corrélation très encourageant de 0,60 entre les notes des correcteurs humains et *LSA*, alors que le *MEV* parvient à un score de 0,37 (tableau 4.16).

Tableau 4.16 : Cotation du corpus de validation sur un espace sémantique général

Copies de validation	Moyenne des deux correcteurs	*LSA*	*MEV*
1	15,5	15,13	14,88
2	16,75	15,17	14,77
3	14,75	15,18	14,28
4	13,75	14,08	13,88
5	17,5	14,44	14,00

6	15	14,37	14,13
7	15,5	14,69	13,75
8	16,5	15,18	15,25
9	15,75	14,75	14,88
10	15,5	14,82	14,81
11	12,75	14,74	14,65
12	17,5	14,81	15,19

Tableau 4.16 : Cotation du corpus de validation sur un espace sémantique général (suite)

Copies de validation	Moyenne des deux correcteurs	*LSA*	*MEV*
13	14,75	14,75	15,00
14	18,25	14,82	14,91
15	10,75	13,81	14,44
16	16,75	15,18	15,22
17	15,25	15,68	14,34
18	15	15,06	15,27
19	16	14,75	14,88
20	14,75	14,81	13,77
21	13,25	14,56	14,06
22	14	14,69	14,93
23	18,5	15,62	15,94
24	13,25	14,56	15,44
25	17,5	15,38	14,71
26	14	14,68	13,76
Corrélation avec les correcteurs :		0,60	0,37

LSA déduit le sens des mots selon leurs contextes. La grande taille du corpus général offre une large quantité d'exemples d'utilisation des mots et aussi un riche vocabulaire pour mieux "comprendre" le sens des mots. En se basant sur ce dernier, *LSA* réussit à mieux juger la qualité des essais et obtient ainsi un meilleur niveau de corrélation qu'avec un petit corpus. Nous utilisons donc uniquement le corpus général dans la suite des expériences.

4.5.3.2 Cotation holistique par classification

La section précédente porte sur l'application de la technique des voisins rapprochés pour coter les essais. Nous explorons ici une alternative similaire à une approche de classification. L'idée principale de la cotation par classification est qu'au lieu de chercher à attribuer une note précise à chaque copie, on peut lui affecter une classe, comme une cote. Si on utilise les copies références pour construire des groupes de qualité proches ou égales, par exemple les bonnes copies dans une classe *A*, les moyennes dans la classe *B* et les mauvaises copies dans une classe *C*, alors on peut juger de la qualité du reste des copies en calculant leurs similarités avec chaque groupe. De cette façon, le problème de cotation sera transformé en un problème de classification.

Nous divisons le corpus référence en trois classes comportant environ le même nombre de cas. Dans la classe *A* on trouve les copies avec une note supérieure à 15, dans la classe *B* ceux entre 13,5 et 15 et dans la classe *C* les copies ayant une note inférieure à 13,5. Par la suite, nous calculons un vecteur moyen par classe représenté dans l'espace sémantique.

Nous suivons les mêmes étapes pour trouver les meilleures performances. Dans chaque espace réduit calculé, on y représente les 25 copies d'entraînement et chaque copie est classifiée selon sa plus grande similarité avec les vecteurs moyens représentatifs des classes.

On évalue la qualité de la classification par le nombre de copies correctement classifiées (*CCC*) c'est-à-dire que la classe attribuée par *LSA* à une copie inclut la notre attribuée par les correcteurs humains pour cette même copie. La figure 4.11 décrit l'évolution des résultats selon le nombre de dimensions. Les résultats du *MEV* sont représentés comme référence et sont constants puisque le nombre de dimensions ne change pas.

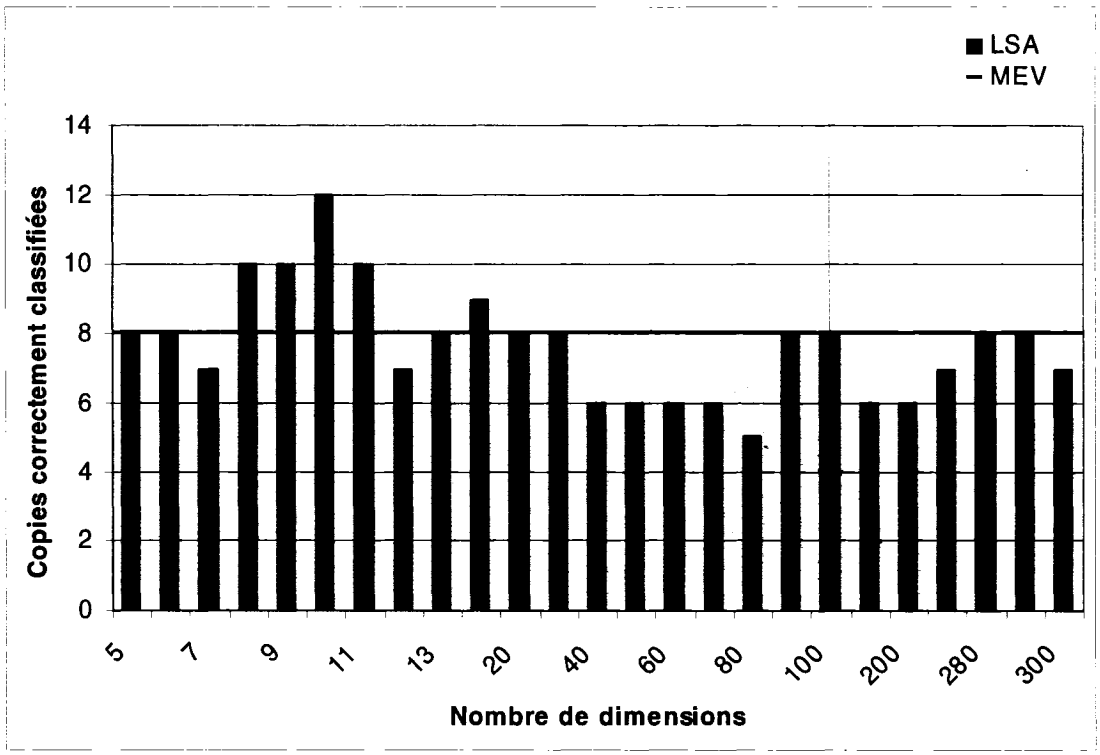

Figure 4.13 : Classification des copies selon le nombre des dimensions

On a constaté que, avec une réduction d'espace sur 10 dimensions, *LSA* réussit à classer correctement 12 copies alors que le *MEV* obtient 7 copies seulement. On a fixé ces valeurs pour les tester avec les copies de validation et les résultats sont encourageants, le *LSA* a obtenu un score de 14 copies bien classées sur 26 ($\chi^2 = 8{,}3$ p=0,08) alors que le *MEV* on a eu 9 de bien classées ($\chi^2 = 3{,}77$ p=0,44)

Le tableau 4.17 affiche les résultats avec un signe « * » à coté des classifications correctes.

Tableau 4.17 : Cotation par classification avec 10 dimensions

Copies de validation	Note humain	classement *LSA*	classement *MEV*
1	15,5	>= 15 *	13,5 < XX < 15
2	16,75	>= 15 *	>= 15 *
3	14,75	<= 13,5	>= 15
4	13,75	>= 15	<= 13,5
5	17,5	>= 15 *	<= 13,5
6	15	13,5 < XX < 15	>= 15 *
7	15,5	>= 15 *	13,5 < XX < 15
8	16,5	>= 15 *	13,5 < XX < 15
9	15,75	<= 13,5	13,5 < XX < 15
10	15,5	<= 13,5	13,5 < XX < 15
11	12,75	>= 15	>= 15
12	17,5	<= 13,5	13,5 < XX < 15
13	14,75	<= 13,5	>= 15
14	18,25	>= 15 *	13,5 < XX < 15
15	10,75	<= 13,5 *	13,5 < XX < 15
16	16,75	>= 15 *	>= 15 *
17	15,25	>= 15 *	>= 15 *
18	15	<= 13,5	>= 15 *
19	16	<= 13,5	>= 15 *
20	14,75	13,5 < XX < 15 *	<= 13,5
21	13,25	>= 15	13,5 < XX < 15
22	14	13,5 < XX < 15 *	13,5 < XX < 15 *
23	18,5	>= 15 *	>= 15 *
24	13,25	>= 15	>= 15
25	17,5	>= 15 *	13,5 < XX < 15
26	14	13,5 < XX < 15 *	13,5 < XX < 15 *
Total des CCC :		14	9

Le tableau 4.9 résume les résultats finaux de la cotation holistique. Les corrélations sont calculées par rapport à la moyenne des notes des deux correcteurs humains. Nous avons obtenu un bon niveau de corrélation selon les voisins les plus proches, comparable à celui de la corrélation inter-juge. Quant à la cotation par classification, *LSA* réussit à en classifier correctement environ la moitié tandis que *MEV* se situe au niveau de la classification au hasard.

Tableau 4.18 : Résumé des résultats de la cotation holistique.

	Cotation selon les voisins les plus proches				Cotation par classification			
	Entraînement		Validation		Entraînement		Validation	
Méthode	*LSA*	*MEV*	*LSA*	*MEV*	*LSA*	*MEV*	*LSA*	*MEV*
Corrélation avec l'humain	0,39	0,21	0,60	0,37	12	7	14 $\chi^2 = 8,3$ p=0,08	9 $\chi^2 = 3,77$ p=0,44

4.5.3 Cotation modulaire

La cotation modulaire, par opposition à la cotation holistique, a pour but d'évaluer la qualité et la pertinence de chaque réponse par rapport à la question. Ce genre de cotation devrait nous fournir plus de détails sur la valeur et la conformité des réponses.

Dans ce qui suit, toutes les copies sont scindées en quatre sections, correspondant aux quatre questions des études de cas. On attribue une note à chaque réponse puis la somme est affectée à la copie. Seul le correcteur 2 a fourni les notes de chaque question séparément, donc dans ce qui suit la corrélation est toujours calculée par rapport au correcteur 2 seulement.

4.5.3.1 Cotation modulaire avec les voisins les plus proches

On utilise les mêmes deux approches, une cotation selon les voisins puis une cotation par classification. Pour chaque technique, on commence par fixer les paramètres de chaque question avec les copies *d'entraînement* puis on les applique aux copies de *validation*.

Pour chaque question, similairement à la cotation holistique, il faut trouver la meilleure dimension de l'espace sémantique et le nombre optimal de voisins à utiliser.

Question 1 :

Avec le corpus d'entraînement, nous avons calculé pour chaque dimension testée la corrélation avec le correcteur en utilisant 1 à 10 voisins. La figure 4.14 représente les courbes des dimensions les plus intéressantes.

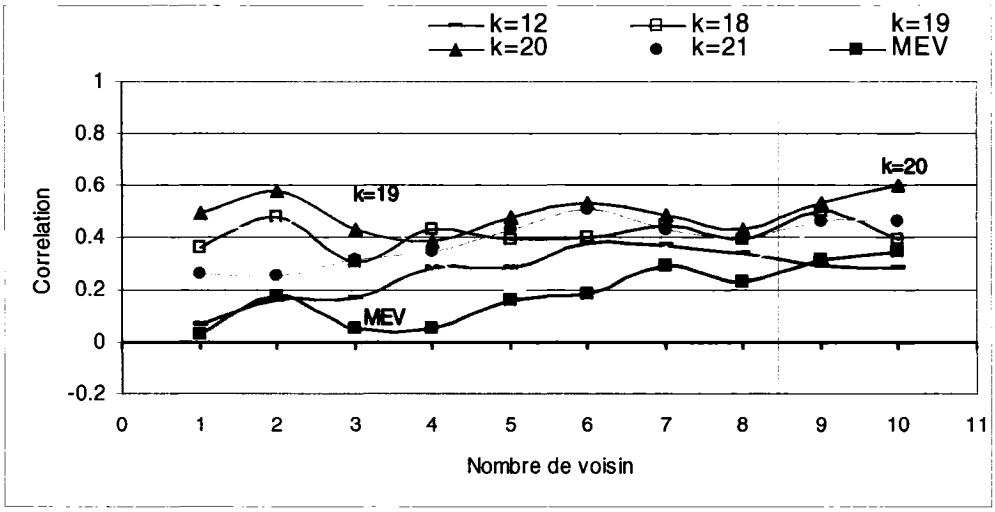

Figure 4.14 : Cotation de la question 1 selon les voisins les plus proches

La meilleure performance est obtenue avec dix voisins sur un espace de 20 dimensions. Ceci correspond à une corrélation de 0,59 pour *LSA* et de 0,34 pour le *MEV*.

Le test de validation sur ces mêmes valeurs des paramètres donne cependant une corrélation négative de -0,08 avec le *LSA* et de 0,34 avec le *MEV*. (Tableau 4.19)

Tableau 4.19 : Cotation de la question 1 selon les plus proches voisins

Copies de validation	Correcteur 2	*LSA*	*MEV*
1	4	3,00	3,51
2	2,5	3,75	3,78
3	1,5	3,00	3,26
4	2,5	3,50	2,74
5	3	3,25	3,25
6	3,5	2,50	3,00
7	3,5	3,00	3,00
8	3,5	3,76	3,75
9	3,5	3,00	2,75
10	2,5	3,25	2,73
11	2	3,00	2,66
12	3	3,25	3,26
13	2	3,25	3,01
14	3,5	3,25	3,25
15	2	3,50	2,75

Tableau 4.19 : Cotation de la question 1 selon les plus proches voisins (suite)

16	3	3,24	3,50
17	3,5	3,75	3,49
18	4	3,50	3,01
19	3,5	3,75	3,26
20	4	3,25	3,50
21	3	3,00	3,24
22	3	3,25	3,00
23	4	3,00	3,24
24	3,5	3,00	3,49
25	3,5	2,75	3,75
26	3,5	3,25	2,76
Corrélation avec le correcteur		-0,08	0,34

Nous répétons les mêmes étapes pour le reste des questions 2, 3 et 4. Les tableaux et les figures associé à chaque question sont insérés dans l'annexe 1 : « Cotation holistique des questions ».

Tableau 4.20 : Corrélation de* LSA *et* MEV *avec les correcteurs humains pour une cotation modulaire

	Corrélation avec les correcteurs humains	
	LSA	*MEV*
Question 1	-0,08	0,34
Question 2	0,35	0,28
Question 3	0,16	-0,08
Question 4	0,18	0,09

Les résultats, constatés tout au long des tests de la cotation des copies de validation par question (tableau 4.9), ont un degré de corrélation au dessous de 0,4. Ceci peut être justifié par la petite taille des réponses; la question 1 a une réponse moyenne de 50 mots, la question 2 a une réponse moyenne de 84 mots, celle de la question 3 contient 45 mots et la réponse moyenne à question 4 se compose de 68 mots. On remarque, effectivement, que le meilleur niveau de corrélation de *LSA* correspond à la plus longue réponse moyenne (question 2).

Les notes finales des copies, étant la somme des notes des questions, *LSA* obtient une corrélation plus proche du correcteur humain que le *MEV* : l'indice de corrélation de *LSA* est de 0,42 contre 0,22 pour le *MEV*. Le tableau 4.12 liste les notes finales des copies de validation.

Tableau 4.21 : les notes finales des copies de validation par cotation holistique

Copies de validation	**Correcteur 2**	***LSA***	***MEV***
1	13,5	15,34	17,35
2	18,5	17,48	17,87
3	15,5	16,00	18,15
4	12	15,55	15,94
5	19	16,41	16,60
6	16	13,70	15,51
7	16	13,63	15,48
8	18	17,25	16,74
9	15,5	14,72	14,91
10	15,5	18,58	17,97
11	13	14,97	15,56
12	18	15,49	15,31
13	16,5	17,09	17,02
14	20,5	17,68	15,17
15	11	14,85	14,84
16	15	15,69	17,04
17	13,5	15,63	16,90
18	16	18,63	15,81
19	17,5	16,34	15,53
20	16,5	15,98	16,49
21	11	14,03	15,87
22	16	16,24	17,65
23	20	15,33	18,14
24	17,5	17,38	16,09
25	19,5	15,58	15,40
26	13,5	15,34	12,62
Corrélation avec le correcteur		0,42	0,22

Le résultat de cette expérience est que *LSA* est meilleure que le *MEV* pour une cotation par question utilisant les voisins les plus proches.

4.5.3.2 Cotation par question selon la classification

Pour chaque question, nous voulions créer trois catégories de réponse selon la qualité. Cependant, à cause du petit intervalle des notes (0 à 4 ou 0 à 6) et la distribution non uniforme de ces derniers, il n'est pas toujours possible ni représentatif de diviser les copies en trois catégories. Par contre une classification sur deux catégories est plus adéquate.

À partir des copies de référence corrigées à notre disposition, nous avons construit, pour chaque question, deux vecteurs représentant les bonnes réponses et les mauvaises réponses. En suite, Les copies d'entraînement sont classifiées dans une des deux catégories selon la similarité de leurs vecteurs.

Pour chaque question, nous utilisons les copies *d'entraînement* pour fixer le nombre de dimensions optimal et nous l'appliquons par la suite au corpus de *validation.*

Question 1 :

La question 1 est notée sur 4 points. La première étape est de construire les deux vecteurs de moyenne. Le premier vecteur est construit à partir des documents de référence ayant une note supérieure à 3 et le deuxième vecteur représente la moyenne du reste des documents.

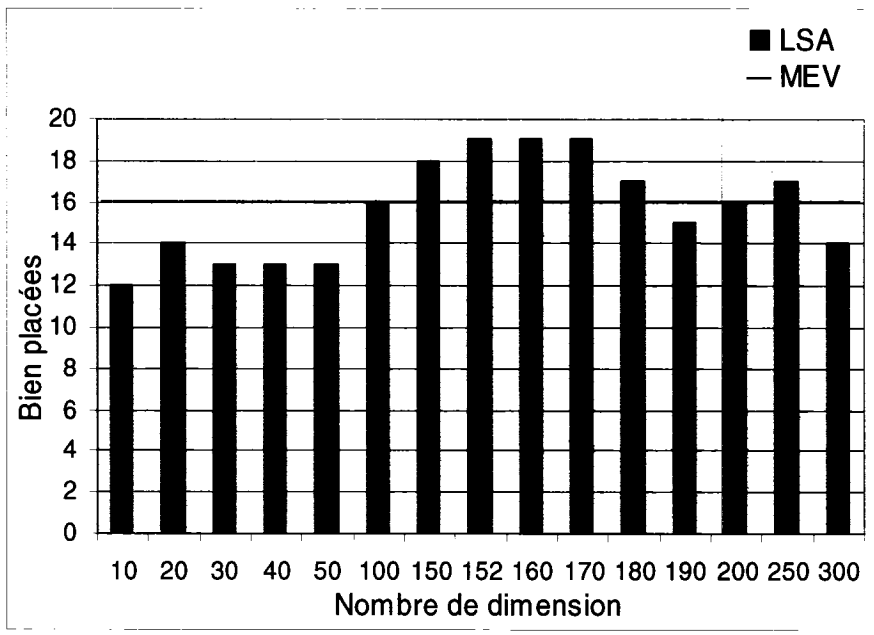

Figure 4.15 : Nombre de copies d'entraînement correctement classifiées pour la question 1 selon les dimensions de l'espace réduit

LSA compte 19 copies bien classées à partir de 152 dimensions alors que le *MEV* obtient 16 copies. En utilisant les copies de validation, *LSA* a classé correctement 10 copies ($\chi^2 = 0{,}23$ p=0,63) et le *MEV* 15 copies ($\chi^2 = 0{,}03$ p=0,86).

Tableau 4.22 : Classification des réponses de la question 1 des copies de validation

Copies de validation	Correcteur 2	*LSA*	*MEV*
1	4	>= 3 *	>= 3 *
2	2,5	>= 3	>= 3
3	1,5	>= 3	>= 3
4	2,5	>= 3	>= 3
5	3	< 3	< 3
6	3,5	< 3	>= 3 *
7	3,5	< 3	< 3
8	3,5	< 3	>= 3 *
9	3,5	< 3	< 3
10	2,5	>= 3	>= 3
11	2	< 3 *	< 3 *
12	3	< 3	>= 3 *

Tableau 4.22 : Classification des réponses de la question 1 des copies de validation (Suite)

13	2	< 3 *	< 3 *
14	3,5	>= 3 *	>= 3 *
15	2	< 3 *	< 3 *
16	3	< 3	>= 3 *
17	3,5	>= 3 *	>= 3 *
18	4	>= 3 *	< 3
19	3,5	< 3	< 3
20	4	< 3	< 3
21	3	< 3	< 3
22	3	< 3	>= 3 *
23	4	< 3	>= 3 *
24	3,5	>= 3 *	>= 3 *
25	3,5	>= 3 *	>= 3 *
26	3,5	>= 3 *	>= 3 *
Total des copies bien classées :		10	15

On répète la même suite d'étapes pour les questions 2,3, et 4. Les détails de ces expériences sont insérés dans l'annexe 2 « cotation par classification – approches modulaire », la figure 4.19 affiche le nombre des copies correctement classifiées par question.

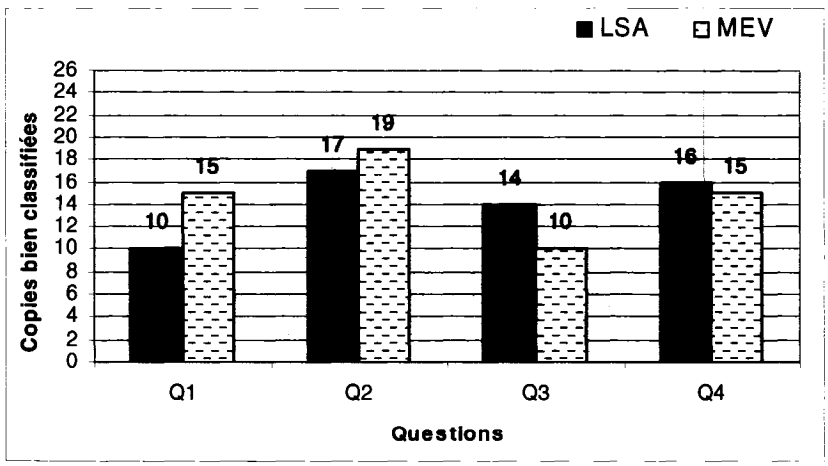

Figure 4.16 : Nombre de copies correctement classifiées avec une cotation modulaire - copies de validation

Les résultats de l'ensemble des tests de classification modulaire ne permettent pas de juger, de manière claire et définitive, de l'efficacité les deux méthodes. Ceci est vraisemblablement dû à la petite taille des réponses n'offrant pas assez d'information pour une classification correcte. Une remarque qui appuie cette hypothèse est que le meilleur taux de classification correcte est obtenu avec la question ayant la plus longue réponse moyenne (Q2 ≈ 84 mots, contre 50, 45 et 68 pour les questions 1, 3 et 4 respectivement). Aussi il faut noter que sur l'ensemble des questions (26 copies × 4 questions = 104 questions), les deux méthodes sont d'une efficacité presque égale : *MEV* obtient un score de 59 copies correctement classifiées contre 57 copies pour *LSA*.

La première expérience de catégorisation ainsi que la deuxième expérience de cotation ont confirmé que *LSA* réussit à détecter et à mesurer la qualité du contenu des copies dans tous les tests où la taille des copies est suffisante.

Comparativement au *MEV*, *LSA* a démontré une grande capacité à détecter le thème de texte et aussi à la qualité de textes.

La classification modulaire n'a pas différencié les deux méthodes. Dans ce dernier cas, les deux méthodes sont d'une efficacité comparable sur l'ensemble des questions.

Conclusion

Dans ce mémoire, nous évaluons la capacité de LSA à extraire le sens de textes dans le cadre d'une correction d'études de cas. Cette technique utilise la décomposition en valeurs singulières qui permet d'éliminer les dimensions sémantiquement insignifiantes et de concentrer les dimensions importantes dans un espace réduit. Nous avons testé LSA pour deux applications basées sur la sémantique de textes.

La première application est une catégorisation de textes où il faut classifier un ensemble de documents selon leurs thèmes. Tandis que la deuxième application est une cotation automatique d'étude de cas pour un devoir de deux pages de textes en écriture libre.

La catégorisation de textes nous a permis de démontrer l'efficacité de LSA et sa grande sensibilité au contenu sémantique des textes. Nous avons comparé LSA au modèle d'espace vectoriel pour une tâche de catégorisation. Les deux techniques réussissent une catégorisation sans fautes, cependant nous avons constaté une grande différence dans la qualité des résultats. En effet, LSA distingue les thèmes avec une grande précision ce qui nous encourage à l'évaluer pour la cotation des essais qui se base sur l'estimation de la qualité du contenu.

La deuxième application est la cotation des essais selon la qualité des réponses. Nous avons utilisé deux approches, la cotation holistique et la cotation modulaire. Pour chaque approche nous avons testé deux techniques, la première est basée sur les voisins les plus proches alors que la deuxième utilise une méthode de classification similaire à la catégorisation.

Pour la cotation holistique, LSA a permit d'approcher la cotation des correcteurs humains mieux que le MEV et ainsi atteindre un niveau de corrélation de 0,6 avec la moyenne des notes des correcteurs. Ce qui est supérieur au niveau de corrélation inter-correcteur (0,44). Quant à la cotation par classification, LSA a correctement notée 53% de copies contre 34% pour le MEV.

Pour la cotation modulaire, les performances de LSA sont supérieures à ceux du MEV dans la plupart des tests. Ainsi pour la cotation selon les voisins les plus proches, LSA obtient une valeur de corrélation de 0,42, ce qui est comparable à la corrélation inter-juge, contre 0,22 pour le MEV. Concernant la cotation par classification, LSA et MEV sont d'une efficacité presque égale et attribuent une cote correcte pour 55% des copies.

Durant ces expériences, nous avons identifié deux points importants. Le premier est que la taille du corpus d'entraînement a un impact direct sur l'efficacité de LSA. Nous avons remarqué qu'un grand corpus améliore considérablement les résultats de la cotation. Le deuxième point est le nombre de dimensions de l'espace réduit. Nous avons constaté que ce paramètre est un élément clé qui doit être rigoureusement choisi durant les tests de calibration.

L'analyse sémantique latente est une technique adéquate pour la cotation des études de cas. LSA a démontré un haut niveau de précision et une capacité remarquable à évaluer la qualité de copies. L'utilisation d'un corpus riche et d'un grand nombre de copies de référence améliorera sans doute les performances de LSA.

Si cette étude démontre un potentiel pour l'application de la technique LSA pour la cotation, une des difficultés pratiques de l'approche étudiée est la nécessité d'avoir un certain nombre de copies d'entraînement déjà cotées. En effet, il est fréquent que les tests et essais comportent du matériel nouveau pour éviter le plagiat et la réutilisation des réponses passées par les étudiants.

Pour contourner ce problème, une solution consiste à utiliser une correction basée sur un solutionnaire. Dans le cadre de cette évaluation, nous n'avons pas étudié rigoureusement le cas d'une évaluation basée sur le solutionnaire. Évaluer l'ensemble des copies par rapport au solutionnaire est une option qui mérite une recherche approfondie pour explorer son application pour une cotation automatique.

La correction automatique des études de cas fait son chemin vers des universités avant-gardistes et sera, peut-être dans un avenir proche, un outil indispensable aussi bien pour les professeurs que pour les étudiants. Le travail réalisé dans le cadre de cette recherche pourrait être un premier pas pour doter l' « École Polytechnique » d'un système de correction automatique adaptable selon le sujet de la matière enseignée. Les travaux futurs peuvent explorer les options non étudiées dans cette évaluation (cotation par rapport au solutionnaire) et aussi améliorer l'interface graphique du programme pour le rendre plus facile à utiliser par le grand public.

Références

[1] : Anderson, J. R., *The adaptive character of thought*, Lawrence Erlbaum Associates, Hillsdale, 1990.

[2] : Andrews, K., *The development of a fast conflation algorithm for English.* Dissertation submitted for the Diploma in Computer Science, University of Cambridge (unpublished), 1971.

[3] : Berry, M.W, Dumais. S.T. & G.W. O'Brien, *Using Linear Algebra for Intelligent Information Retrieval*, page 24, Decembre 1994

[4] : Burstein.J, Chodorow.M, *Beyond Essay Length: Evaluating e-rater's Performance on TOEFL Essays*, pages 4-8, Février 2004.

[5] : Cooper, W. S., *Some inconsistencies and misnomers in probabilistic information retrieval.*, ACM Transactions on Information Systems (TOIS), v.13 n.1, pages 100-111, Jan. 1995.

[6] : Dumais, S.T, *Enhancing performance in latent semantic indexing (LSI) retrieval, Technical Report*, Bellcore, page 19, 1990.

[7] : Frakes, William B., *Term Conflation for Information Retrieval*, SIGIR, pages 383-389, 1984.

[8] : G. Furnas, S. Deerwester, S. Dumais, T. Landauer, R. Harshman, L. Streeter and K. Lochbaum, *Information retrieval using a singular value decomposition model of latent semantic structure*, ACM Press, pages 465-480, 1988.

[9] : Greengrass, Ed., *Information Retrieval: A Survey*, (2000)

[10] : Kraaij, W., Pohlmann, R., *Porter's stemming algorithm for Dutch*, Informatiewetenschap, pages 167-180, 1994.

[11] : Krovetz, R., *Viewing Morphology as an Inference Process*, Proceedings of the Sixteenth Annual International Conference on Research and Development in Information Retrieval, pages 191-203, 1993.

[12] : Landauer, T. K. & Dumais, S. T., *A solution to Plato's problem: The Latent Semanctic Analysis theory of the acquisition, induction, and representation of knowledge*, Psychological Review, 104, pages 211-140, 1997.

[13] : Landauer, T. K., Foltz, P. W., et Laham, D., *Introduction to Latent Semantic Analysis*, Discourse Processes, vol.25, pages 259-284, 1998.

[14] : Landauer.T. K., Laham, D., Foltz, P. W., *The Intelligent Essay Assessor*, IEEE Intelligent Systems, vol.15, n.5, pages 27-31, 2000.

[15] : Lovins, B.J., *Development of a stemming algorithm. Mechanical Translation and Computational Linguistics*, vol.11, pages 22-31 (1968).

[16] : Luhn, H. P., *A Statistical Approach to Mechanized Encoding and Searching of Literary Information*, American Chemical Society meeting in Miami, April , 1957.

[17] : Luhn, H.P., *The automatic creation of literature abstracts*, IBM Journal of Research and Development, 2, pages 159-165, Avril 1958.

[18] : Maron, M., Kuhns, J., *On relevance, probabilistic indexing, and information retrieval*. Journal of the ACM 7, pages 216-244, 1960.

[19] : Miller, T., *Essay Assessment with Latent Semantic Analysis*, Journal of Educational Computing Research, vol.28, n.3, pages 7-24, 2003.

[20] : Niedermair, G.T., Thurmair, G., Biittel, I., *A retrieval tool on the basis of morphological analysis*, Research and development in information retrieval, pages 369-380, Mars 1985.

[21] : Page, E. B., *The imminence of grading essays by computer*, Phi Delta Kappan, vol.47, pages 238-243, 1966.

[22] : Popovie, M., Willett, P., *The Effectiveness of Stemming for Natural-Language Access to Slovene Textual Data*, Journal of the American Society for Information Science, Vol.43, pages 384-390, 1992.

[23] : R Development Core Team, *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, URL www.R-project.org; 2004.

[24] : Rehder, B., M.E. Schreiner, M.B. Wolfe, D. Laham, T.K. Landauer, and W. Kintsch. *Using Latent Semantic Analysis to assess knowledge: Some technical considerations*. Discourse Processes 25, pages 337-354, 1998.

[25] : Rijsbergen, C. J., *Information retrieval* 2ème édition, Butterworths, 1979.

[26] : Robertson, S. E. *Specificity and weighted retrieval*. Journal of Documentation, n 30, pages 41- 46, 1974.

[27] : Robertson, S. E., Maron, M. E., Cooper, W. S., *Probability of relevance: A unification of two competing models for document retrieval.* Information Technology: Research and Development, l(l): pages 1-21; 1982.

[28] : Robertson, S. E., Sparck Jones, K., *Relevance weighting of search terms.* Journal of the American Society for Information Science, 27(3), pages 129-146, 1976.

[29] : Robertson, S. E., *The probability ranking principle in IR*, Journal of Documentation, 33(4), pages 294–304, December 1977.

[30] : Salton, G., A. Wong & C. S. Yang., *A vector space model for automatic indexing*, Communications of ACM, vol. 18, pages 613 - 620, 1975.

[31] : Salton, G., Fox, E. A., Wu, H., *Extended Boolean information retrieval*, Communications du ACM, vol.26 n.11, pages 1022–1036, 1983.

[32] : Stein, A., Schmid, H., *Étiquetage morphologique de textes français avec un arbre de décisions*, page 13, 1995.

[33] : Wolfe, M. B., Schreiner, M. E., Rehder, B., Laham, D., Foltz, P. W., Kintsch, W., & Landauer, T. K., *Learning from text: Matching readers and text by Latent Semantic Analysis*, Discourse Processes, vol.25, pages 309-336, 1998.

[34] : Wresch, W., *The Imminence of Grading Essays by Computer--25 Years Later*, Computers and composition, vol.10, n.2, pages 45-58, April 1993.

Annexes

Annexe A : Cotation modulaire selon les voisins les plus proches

Dans chapitre 4, on a commencé à présenter les résultats de la cotation modulaire, on reprend ici à partir de la question 2.

A.1 Question 2 :

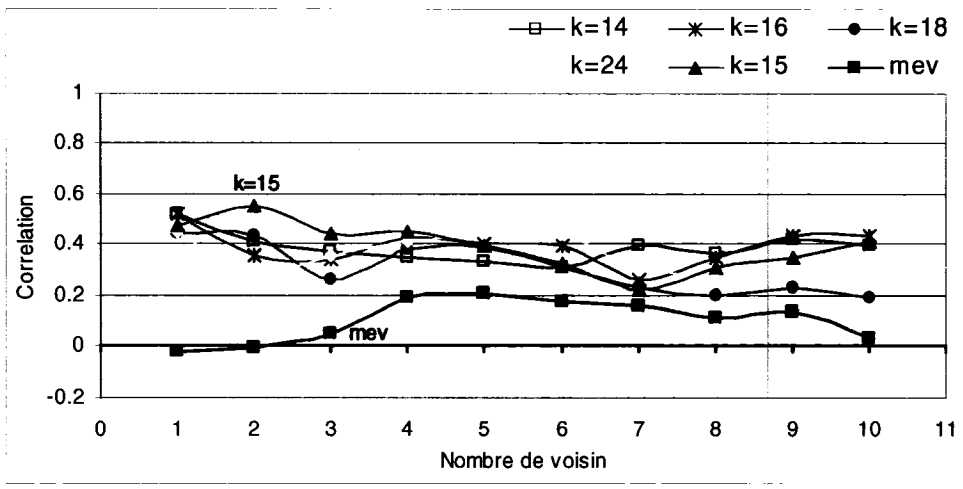

Figure a.1 : La cotation de la question 2 selon les voisins les plus proches

Pour les copies d'entraînement, la meilleure corrélation est obtenue en considérant les deux voisins les plus proches sur un espace de 15 dimensions. Ce qui donne une corrélation de 0,55 pour *LSA* et 0,002 pour le *MEV*.

La cotation des copies de validation avec *LSA* dans les mêmes conditions donne une corrélation de 0,35 avec le correcteur humain alors que *MEV* a une valeur de 0,28.

Tableau a.1 : Cotation de la question 2 selon les plus proches voisins

Copies de validation	Humain	*LSA*	*MEV*
1	3	4,25	4,78
2	6	5,00	5,00
3	5	4,75	5,51
4	3,5	4,25	4,52
5	5	4,75	4,50
6	4	3,74	4,25
7	5	3,48	4,27
8	5	4,75	4,50
9	4	4,00	4,25
10	4,5	5,75	5,52
11	4	4,00	4,52
12	5	4,25	4,22
13	5	5,00	5,25
14	6	5,24	4,01
15	3	3,74	4,00
16	3,5	4,26	4,75
17	4	4,00	4,50
18	3	5,50	4,24
19	5,5	4,24	4,24
20	4	4,50	4,47
21	3	3,74	4,50
22	5	4,75	5,49
23	6	4,25	5,50
24	5	5,25	4,26
25	5,5	4,75	4,00
26	3,5	4,25	3,02
Corrélation avec le correcteur :		0,35	0,28

A.2 Question 3 :

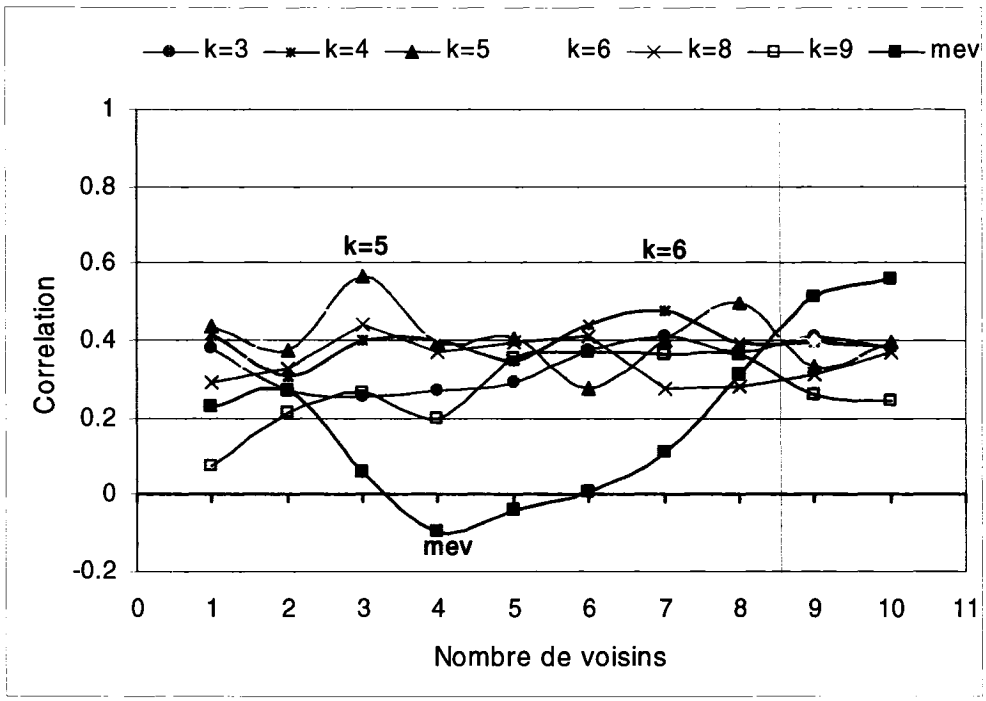

Figure a.2 : La cotation de la question 3 selon les voisins les plus proches

La meilleure corrélation est obtenue avec 5 dimensions (k=5) et trois voisins les plus proches. Avec les copies d'entraînement, cela correspond à 0,56 pour *LSA* et 0,06 pour le *MEV*. Concernant les copies de validation, on a obtenu une corrélation de 0,16 avec *LSA* et -0,08 pour le *MEV*. (Tableau 1.2)

Tableau a.2 : Cotation de la question 3 selon les plus proches voisins

Copies de	Humain	*LSA*	*MEV*
1	3,5	2,66	3,03
2	2	3,00	3,00
3	3	3,17	3,66
4	3	3,67	3,00
5	3	2,32	3,34
6	4	3,50	3,17
7	2	2,67	3,37
8	3	2,68	2,66
9	3	3,83	3,02
10	2	3,50	3,32
11	2,5	2,50	2,34
12	3	3,17	3,47
13	2	3,33	2,67
14	3	3,17	3,35
15	1,5	3,17	3,54
16	3	3,16	3,49
17	3	3,17	3,66
18	2	2,67	2,50
19	3	2,82	3,67
20	3	3,83	2,84
21	2,5	3,00	3,10
22	3	2,50	2,67
23	3	2,67	2,63
24	1	2,99	3,35
25	2	3,00	3,67
26	1	2,67	3,31
Corrélation avec le correcteur :		0,16	-0,08

A.3 Question 4 :

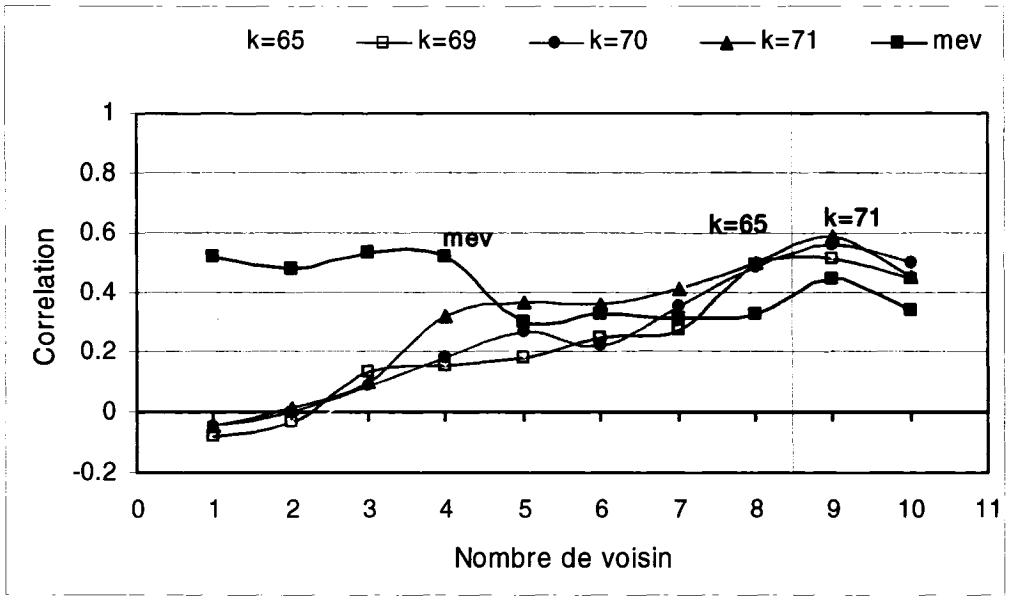

Figure a.3 : La cotation de la question 4 selon les voisins les plus proches

La corrélation maximale obtenue est avec k=71 et neuf voisins les plus proches. C'est-à-dire une corrélation de 0,58 pour *LSA* et de 0,32 pour le *MEV*. Concernant les copies témoins, on a obtenu une corrélation de 0,18 avec *LSA* et 0,09 pour le *MEV*. La liste des notes est présentée dans le tableau 4.11.

Tableau a.3 : Cotation de la question 4 selon les plus proches voisins

Copies de validation	Humain	*LSA*	*MEV*
1	3,5	3,84	4,28
2	4	3,73	4,10
3	4	3,50	3,86
4	2,5	3,55	4,17
5	6	3,67	4,35
6	4,5	3,72	4,01
7	2,5	3,68	3,94
8	4,5	3,99	3,99
9	4	3,72	3,66
10	4	3,83	4,21
11	3	3,96	3,87
12	5	3,74	3,62
13	4,5	3,82	3,50
14	5	3,95	3,91
15	3	3,86	4,09
16	5	3,94	4,05
17	2	3,89	4,41
18	6	4,14	4,32
19	3	4,11	3,80
20	4,5	3,72	4,06
21	2	3,55	3,63
22	3	3,50	3,66
23	4	3,84	3,89
24	4	3,88	4,09
25	5	3,33	3,66
26	3	3,60	3,82
Corrélation avec le correcteur :		0,18	0,09

Annexe B : Cotation modulaire par classification

B.1 Question 2 :

Pour la question 2, les copies sont notées sur 6 points. Le vecteur des meilleures copies se compose de ceux ayant une note supérieure à 4,5.

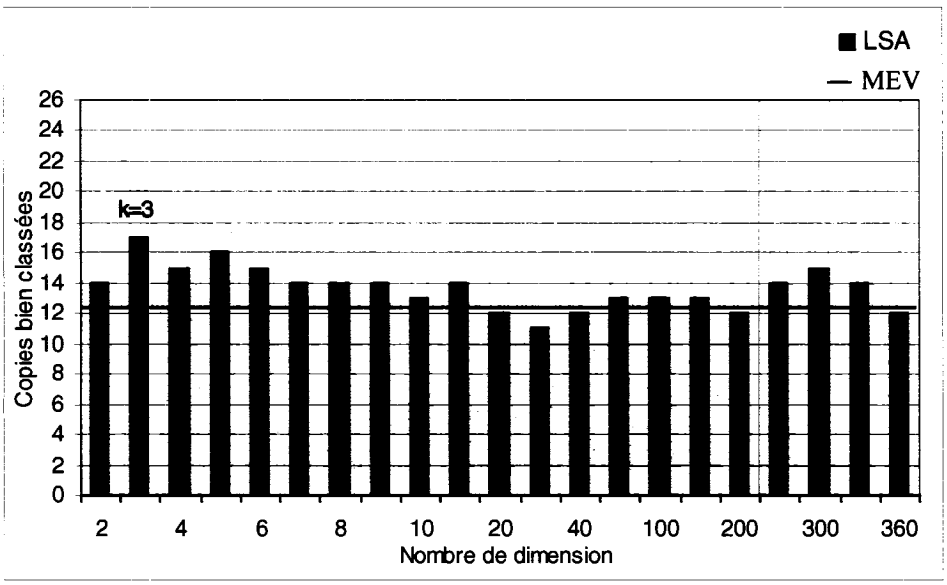

Figure b.1 : Nombre de copies d'entraînement correctement classifiées pour la question 2 selon les dimensions de l'espace réduit

Sur l'ensemble des dimensions testées, *LSA* obtient des résultats meilleurs ou égaux à ceux du *MEV*. On remarque qu'avec une projection sur trois dimensions *LSA* a 17 copies correctement classifiées alors que *MEV* n'a classifié que 11 copies.

On a appliqué la même projection sur 3 dimensions pour les copies témoins, le *LSA* classifie correctement 17 copies alors que le *MEV* réussit à classifier 19 copies.

Tableau b.1 : Classification des réponses de la question 2 des copies de validation

Copies de validation	Humain	*LSA*	*MEV*
1	3	>= 4,5	>= 4,5
2	6	< 4,5	>= 4,5 *
3	5	>= 4,5 *	>= 4,5 *
4	3,5	< 4,5 *	>= 4,5
5	5	< 4,5	< 4,5
6	4	< 4,5 *	< 4,5 *
7	5	< 4,5	< 4,5
8	5	< 4,5	>= 4,5 *
9	4	< 4,5 *	< 4,5 *
10	4,5	>= 4,5 *	>= 4,5 *
11	4	>= 4,5	< 4,5 *
12	5	< 4,5	>= 4,5 *
13	5	>= 4,5 *	>= 4,5 *
14	6	>= 4,5 *	< 4,5
15	3	< 4,5 *	< 4,5 *
16	3,5	< 4,5 *	< 4,5 *
17	4	< 4,5 *	< 4,5 *
18	3	>= 4,5	>= 4,5
19	5,5	< 4,5	< 4,5
20	4	< 4,5 *	< 4,5 *
21	3	< 4,5 *	< 4,5 *
22	5	>= 4,5 *	>= 4,5 *
23	6	>= 4,5 *	>= 4,5 *
24	5	>= 4,5 *	>= 4,5 *
25	5,5	>= 4,5 *	>= 4,5 *
26	3,5	< 4,5 *	< 4,5 *
Total des copies correctement classifiées		17	19

B.2 Question 3 :

La question 3 est notée sur un intervalle de 0 à 4 points. Le premier vecteur est la moyenne des copies références ayant une note supérieure à 3 et le deuxième vecteur représente la moyenne du reste des documents.

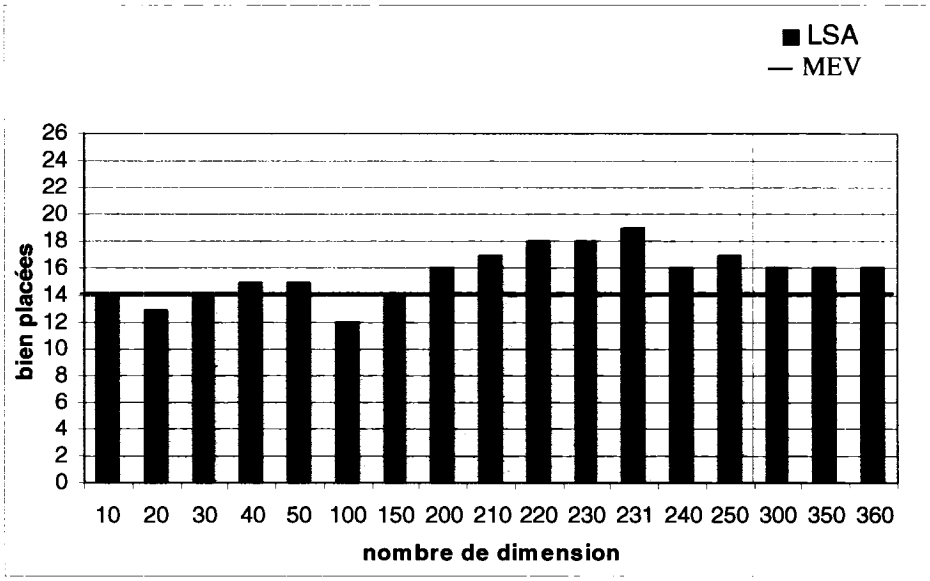

Figure b.2 : Nombre de copies d'entraînement correctement classifiées pour la question 3 selon les dimensions de l'espace réduit

D'après le test avec les copies d'entraînement, on a trouvé qu'à 231 dimensions, *LSA* classifie correctement 19 copies contre 14 pour le *MEV*.

Pour les copies cibles, sur 231 dimensions, *LSA* compte 14 copies bien placées contre 10 pour le *MEV*.

Tableau b.2 : Classification des réponses de la question 3 des copies de validation

Copie de validation	Humain	*LSA*	*MEV*
1	3,5	< 3	>= 3 *
2	2	>= 3	>= 3
3	3	< 3	>= 3 *
4	3	< 3	< 3
5	3	< 3	>= 3 *
6	4	< 3	>= 3 *
7	2	>= 3	>= 3
8	3	< 3	< 3
9	3	>= 3 *	>= 3 *
10	2	< 3 *	>= 3
11	2,5	>= 3	< 3 *
12	3	< 3	>= 3 *
13	2	< 3 *	>= 3
14	3	< 3	>= 3 *
15	1,5	< 3 *	>= 3
16	3	>= 3 *	>= 3 *
17	3	>= 3 *	>= 3 *
18	2	< 3 *	>= 3
19	3	>= 3 *	< 3
20	3	>= 3 *	< 3
21	2,5	< 3 *	>= 3
22	3	>= 3 *	< 3
23	3	>= 3 *	< 3
24	1	< 3 *	>= 3
25	2	>= 3	>= 3
26	1	< 3 *	>= 3
Total des copies bien classifiées		14	10

B.3 Question 4 :

La question 4 est notée sur 6 points, la moitié des copies ont une note supérieure à 4 points. On représente les deux vecteurs moyens et les comparent à chaque copie d'entraînement.

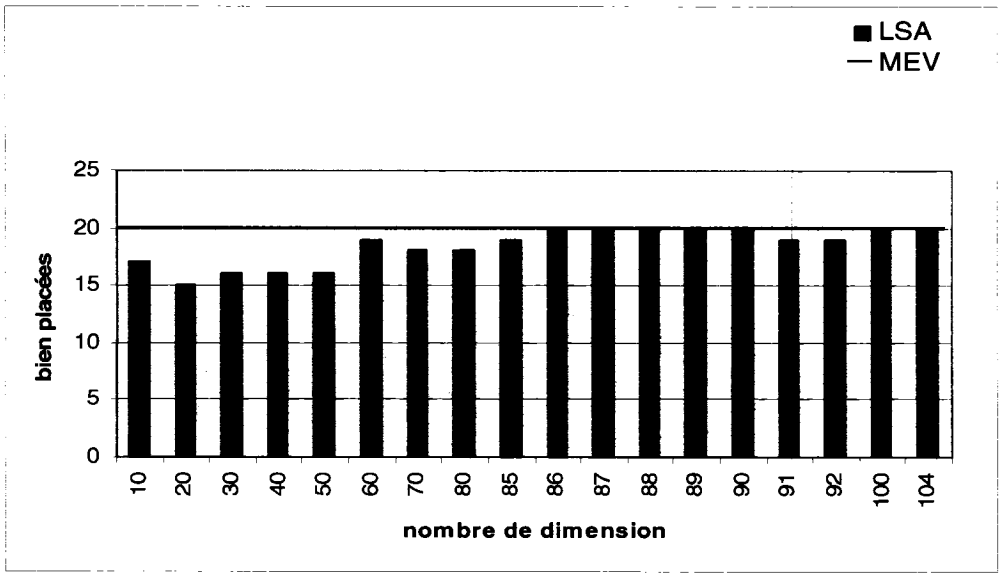

Figure b.3 : Nombre de copies d'entraînement correctement classifiées pour la question 4 selon les dimensions de l'espace réduit

Généralement, *MEV* retourne des résultats supérieurs ou égaux à ceux du *LSA*, ainsi le meilleurs cas pour *LSA* est 20 copies bien placées avec 86 dimensions.

On classifie les copies de validation avec *LSA* et *MEV* sur un espace de 86 dimensions, *LSA* obtient un score de 16 et *MEV* obtient 15.

Tableau b.3 : Classification des réponses de la question 4 des copies de validation

Copie de validation	Humain	*LSA*	*MEV*
1	3,5	>= 4	>= 3 *
2	4	>= 4 *	>= 3 *
3	4	< 4	>= 3 *
4	2,5	>= 4	>= 3
5	6	>= 4 *	>= 3 *
6	4,5	< 4	>= 3 *
7	2,5	< 4 *	>= 3
8	4,5	>= 4 *	< 3
9	4	< 4	< 3
10	4	< 4	>= 3 *
11	3	< 4 *	>= 3 *
12	5	< 4	< 3
13	4,5	>= 4 *	< 3
14	5	< 4	>= 3 *
15	3	< 4 *	< 3
16	5	>= 4 *	>= 3 *
17	2	>= 4	>= 3
18	6	>= 4 *	>= 3 *
19	3	< 4 *	< 3
20	4,5	>= 4 *	>= 3 *
21	2	< 4 *	< 3 *
22	3	< 4 *	< 3
23	4	>= 4 *	>= 3 *
24	4	>= 4 *	>= 3 *
25	5	< 4	< 3
26	3	< 4 *	>= 3 *
Total des copies bien classifiées		16	15